Inhalt

Zielsetzung

Migration der Windows Server und der Exchange Server

Meine Infrastruktur soll auf Windows Server 2019 aktualisiert werden. In diesem Abschnitt der Umstellung sind meine beiden Exchange Server 2016 dran. Beide laufen als virtuelle Maschine auf je einem Hyper-V-Host.

Mit Windows Server 2019 als Betriebssystem kann ich gleichzeitig auf Exchange Server 2019 migrieren.

Die Migration wird durch ein Wipe & Load je Server durchgeführt. Dabei deinstalliere ich jeweils einen Exchange Server, entferne das alte Betriebssystem, installiere einen neuen Windows Server 2019 und installiere darauf den neuen Exchange Server.

Wichtig ist mir dabei, dass der Mailservice ohne Unterbrechung weiterläuft. Die fehlende Hochverfügbarkeit während der Umstellung kann ich akzeptieren.

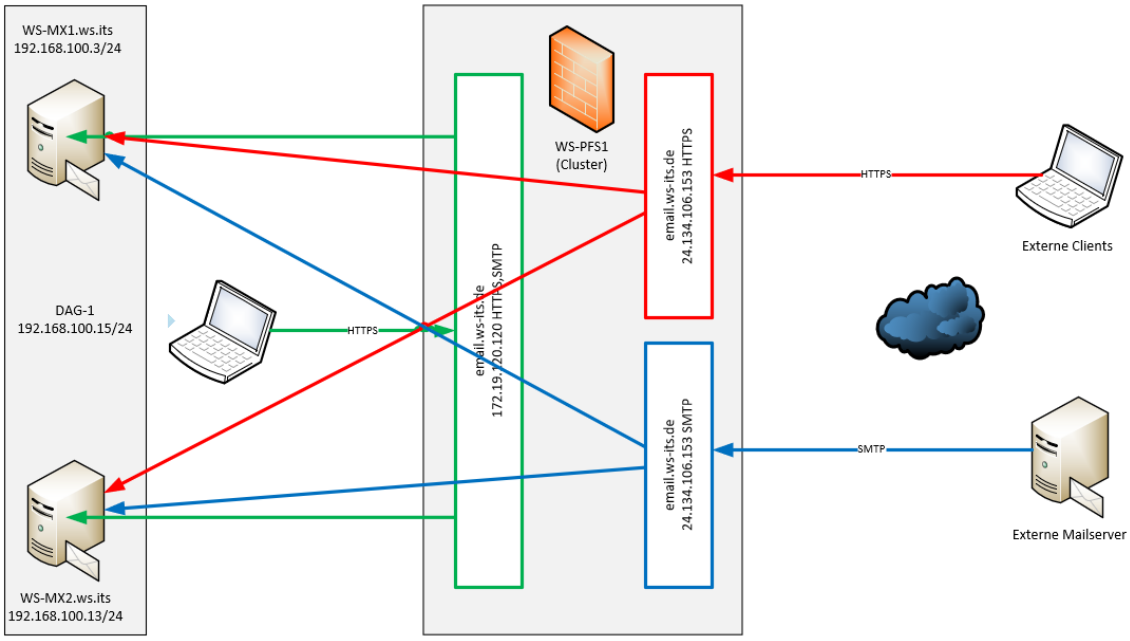
Der Mailservice

Beide Mailserver stellen meine Mailboxen mit insgesamt 4 Datenbanken zur Verfügung. Jeder der Server hält eine Kopie der 4 Datenbanken in einer Datenbankverfügbarkeitsgruppe (DAG). Der für diesen Cluster erforderliche Zeugenserver ist mein WS-FS3. Dieser steht im Außenstandort.

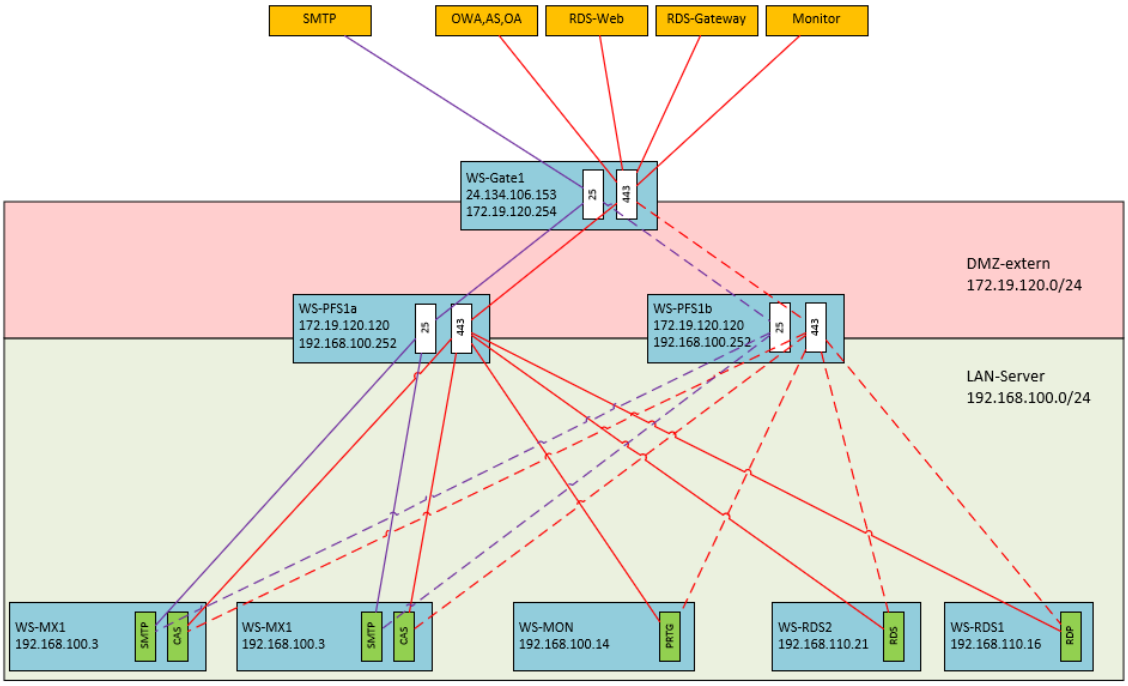
Die Clients greifen intern wie extern über den Namespace email.ws-its.de zu. Beide Mailserver verwenden dafür ein externes Webserver-Zertifikat. Der Zugriff wird über einen Loadbalancer gesteuert. Dieser läuft auf meinen beiden virtuellen PFSense-Servern mit dem Service HAProxy. Beide PFSense-Systeme bilden einen Cluster.

Durch den HAProxy werden auch externe Mails an meine beiden Mailserver zugestellt.

Das ist meine Mailserver-Infrastruktur:



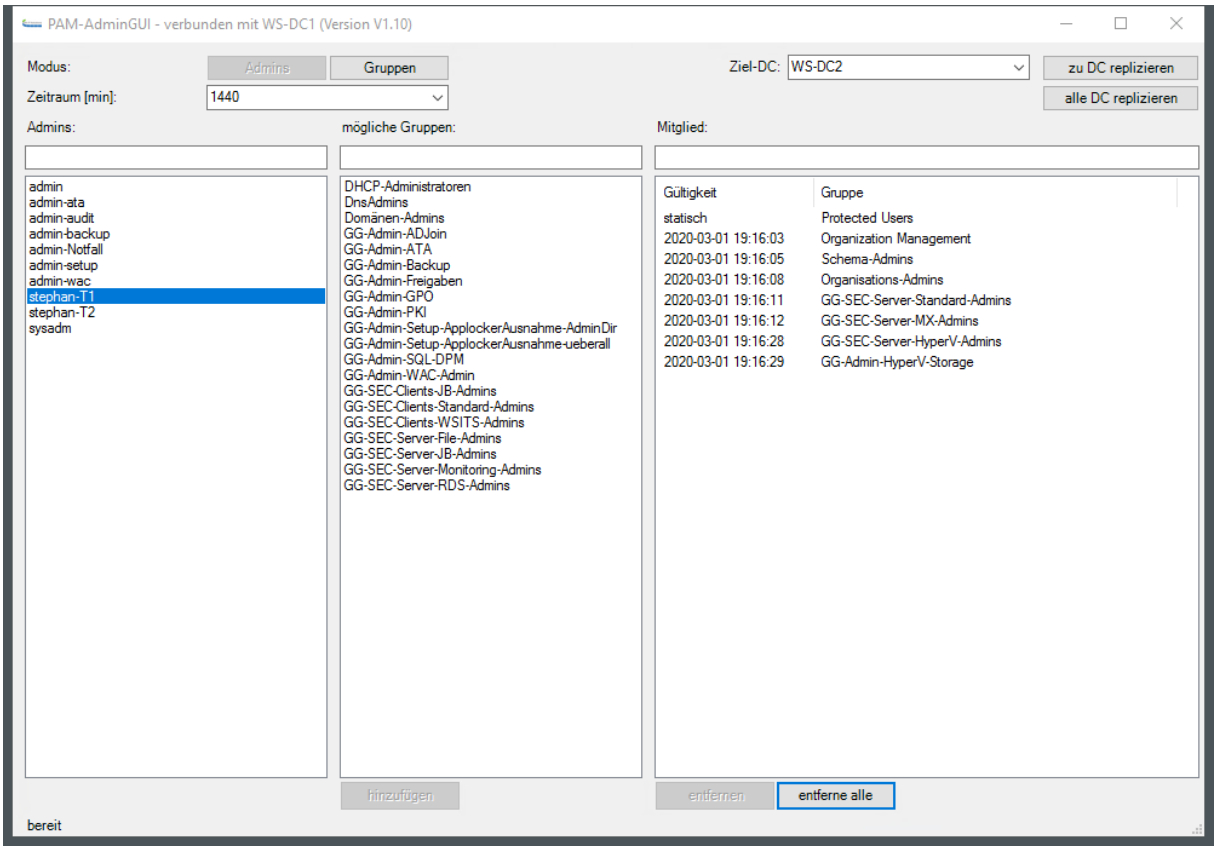
Den Loadbalancer erkennt man in dieser Grafik besser:



Der Mailserver WS-MX2 hat Probleme mit der Datenbank-Bereitstellung. Daher werde ich diesen zuerst neu installieren.

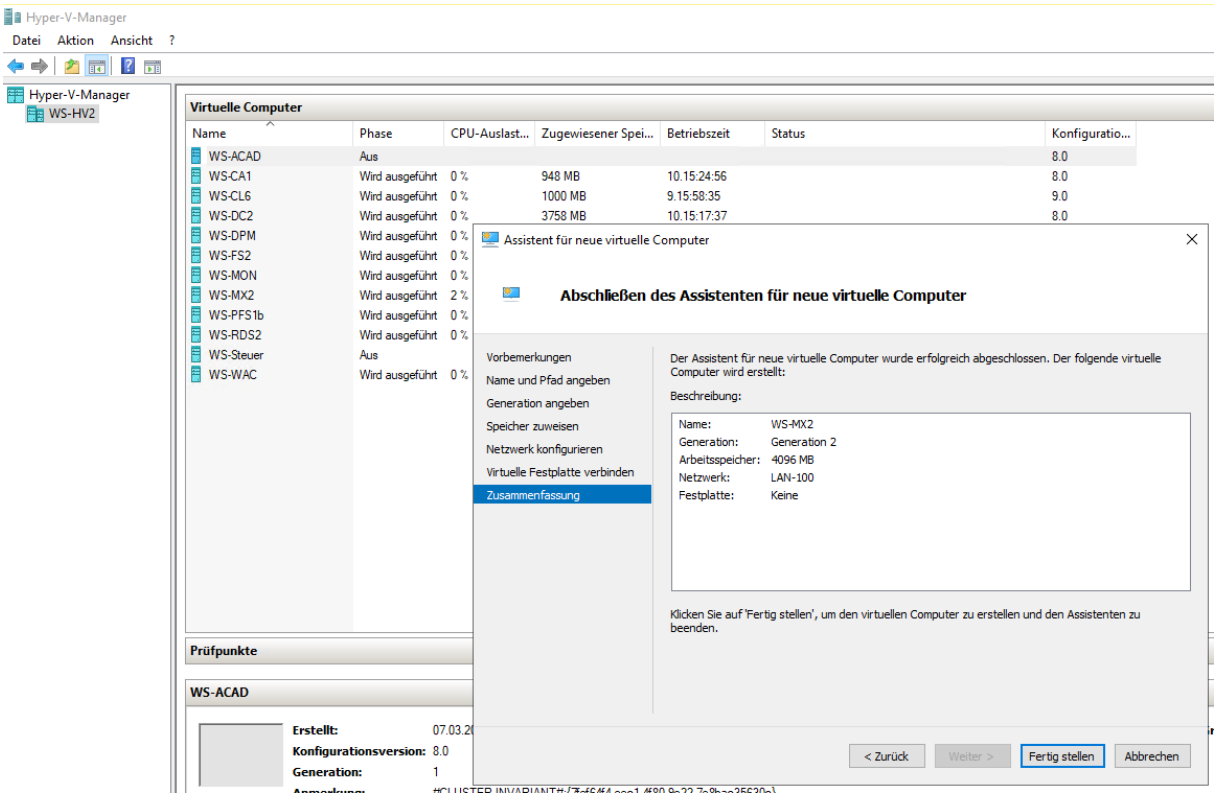
Vorbereitung

Ich verwende in meiner Infrastruktur ein Tier-Modell für die Administration und zusätzlich eine zeitliche Begrenzung von Gruppenmitgliedschaften. Gesteuert werden diese durch meine eigene PAM-Scriptlösung (Privileged Access Management). Ich weise meinem Account für die Server-Administration (T1) die erforderlichen Gruppenmitgliedschaften zu:

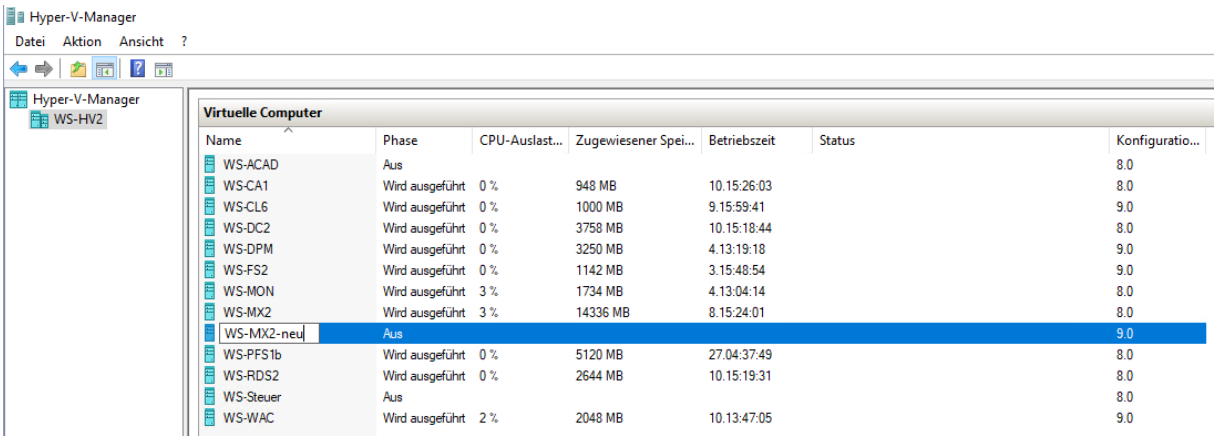


Aufbau der neuen VM

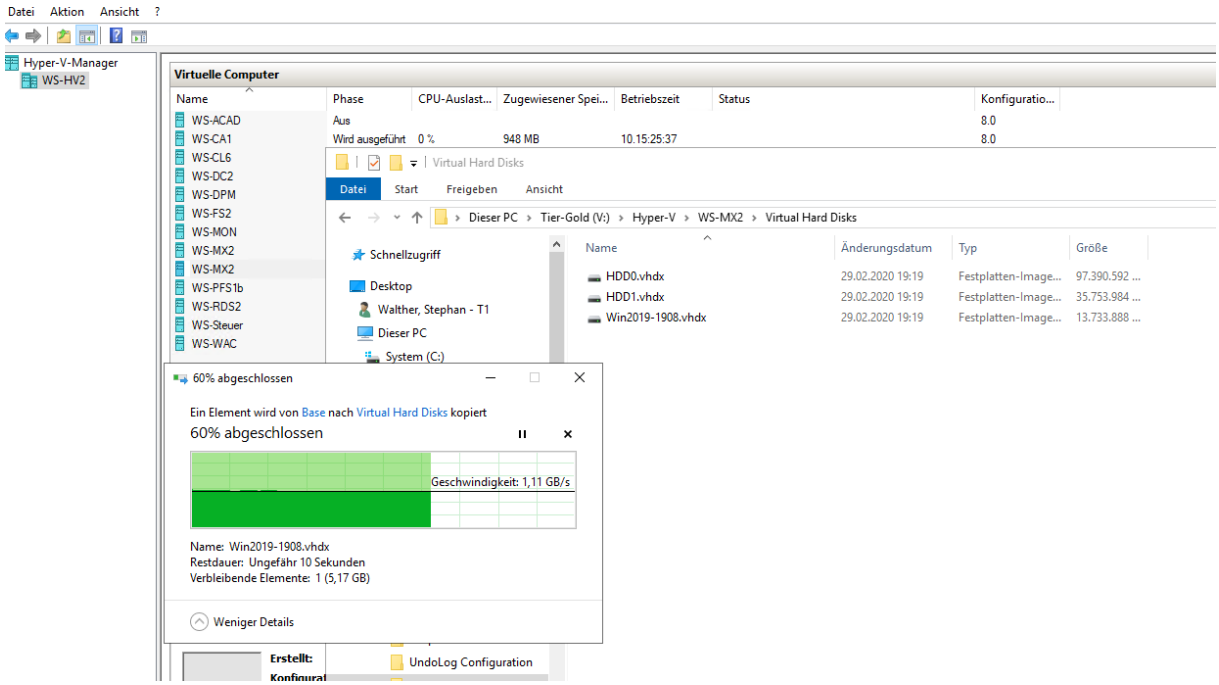
Mit diesen Rechten ausgestattet melde ich mich an meinem Hyper-V-Host an, auf dem der aktuelle WS-MX2 läuft. Hier erstelle ich eine neue virtuelle Maschine. Die Festplatte lasse ich weg:



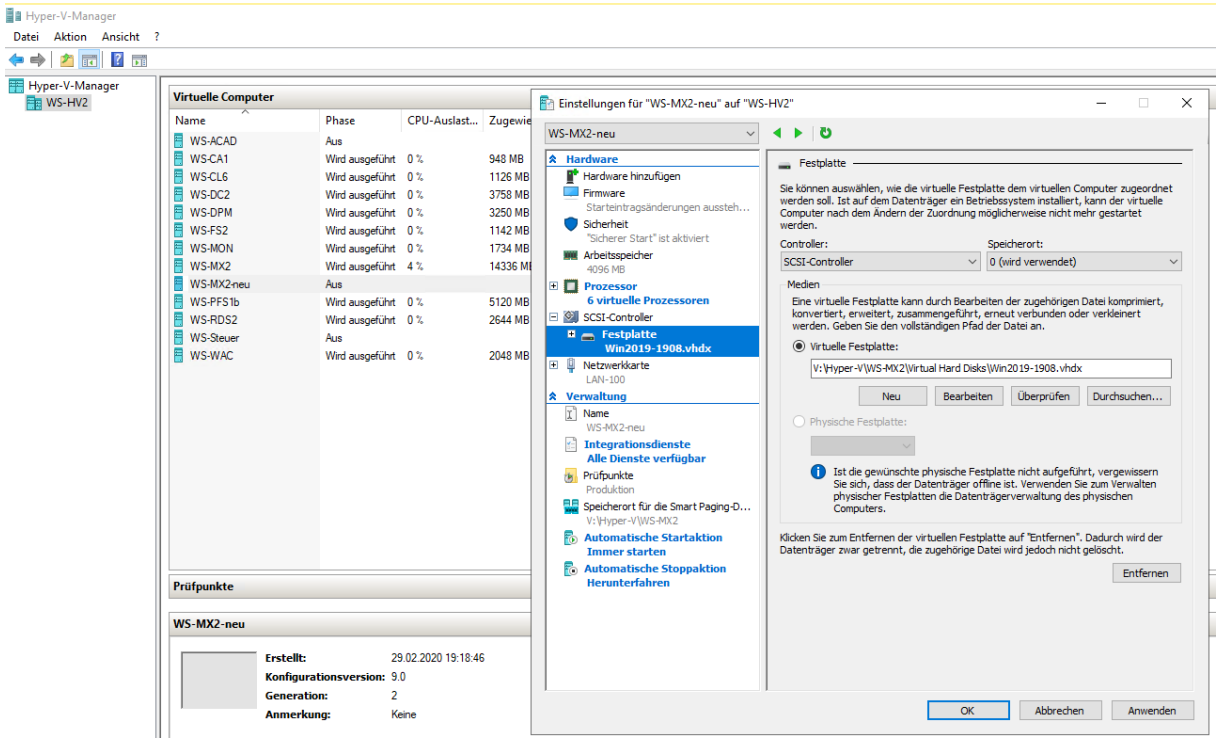
Den Server habe ich bereits mit dem neuen Namen erstellt. Damit passt dann auch der Pfad im Hyper-V-Volume. Damit es später keine Verwechslung gibt, ändere ich den Anzeigenamen:



Jetzt kopiere ich meine Basefile in das Verzeichnis der neuen VM. Darin ist ein installierter und (einigermaßen) aktueller Windows Server 2019 mit grafischer Oberfläche installiert. Das Betriebssystem hatte ich mit sysprep zurückgesetzt und generalisiert:

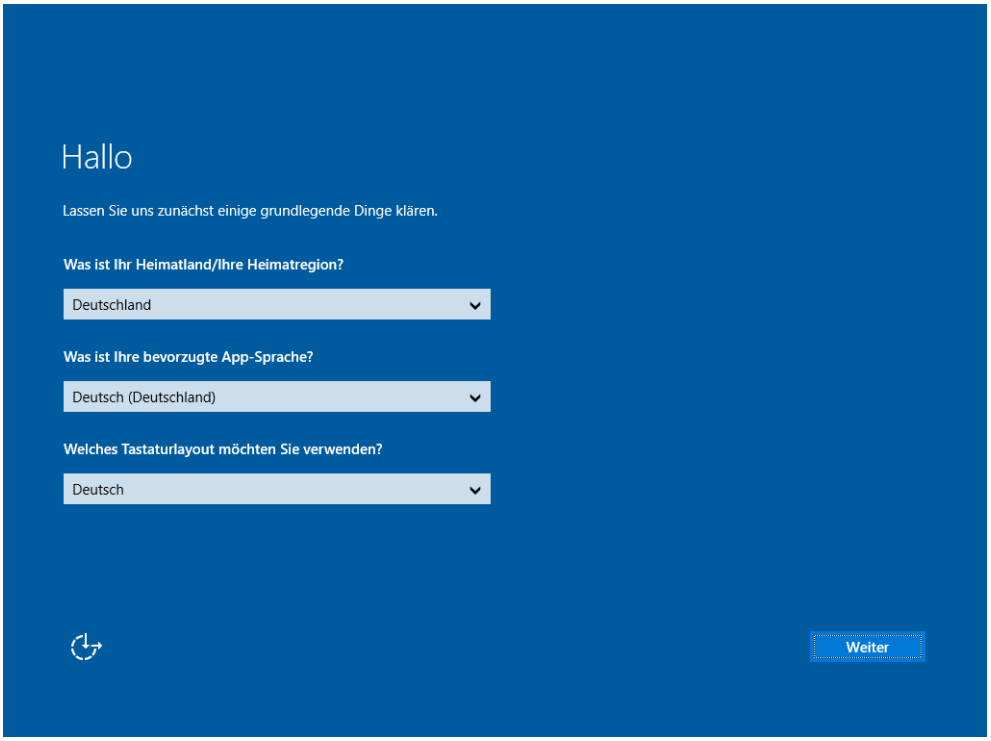


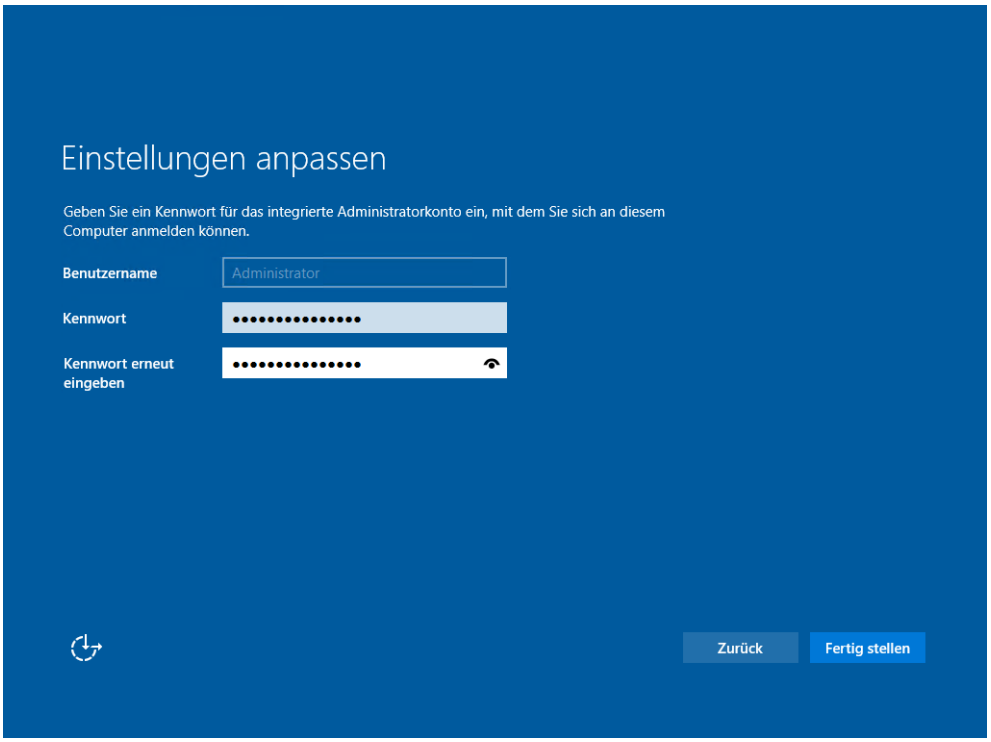
Jetzt binde ich die kopierte VHDX-Datei in die neue VM ein. Zusätzlich ändere ich noch ein paar andere Parameter wie die Anzahl der CPU-Kerne und die Integrationsdienste. Auch die Firmware-Starteinstellung wird modifiziert. Das geht aber erst nach dem Einbau der VHDX:



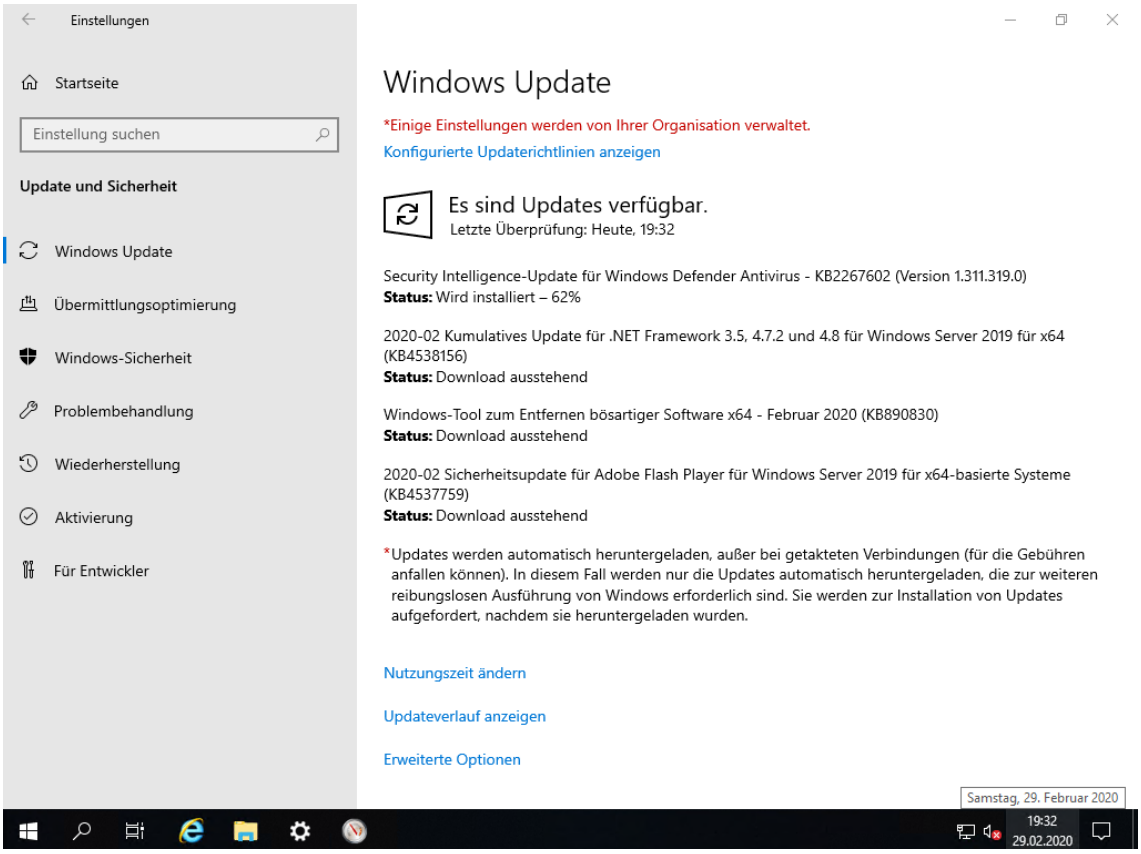
Bereitstellung des neuen Betriebssystems

Im nächsten Schritt starte ich die VM. Das Betriebssystem führt die Out-Of-Box-Experience (OOBE) aus und startet die Einrichtung:



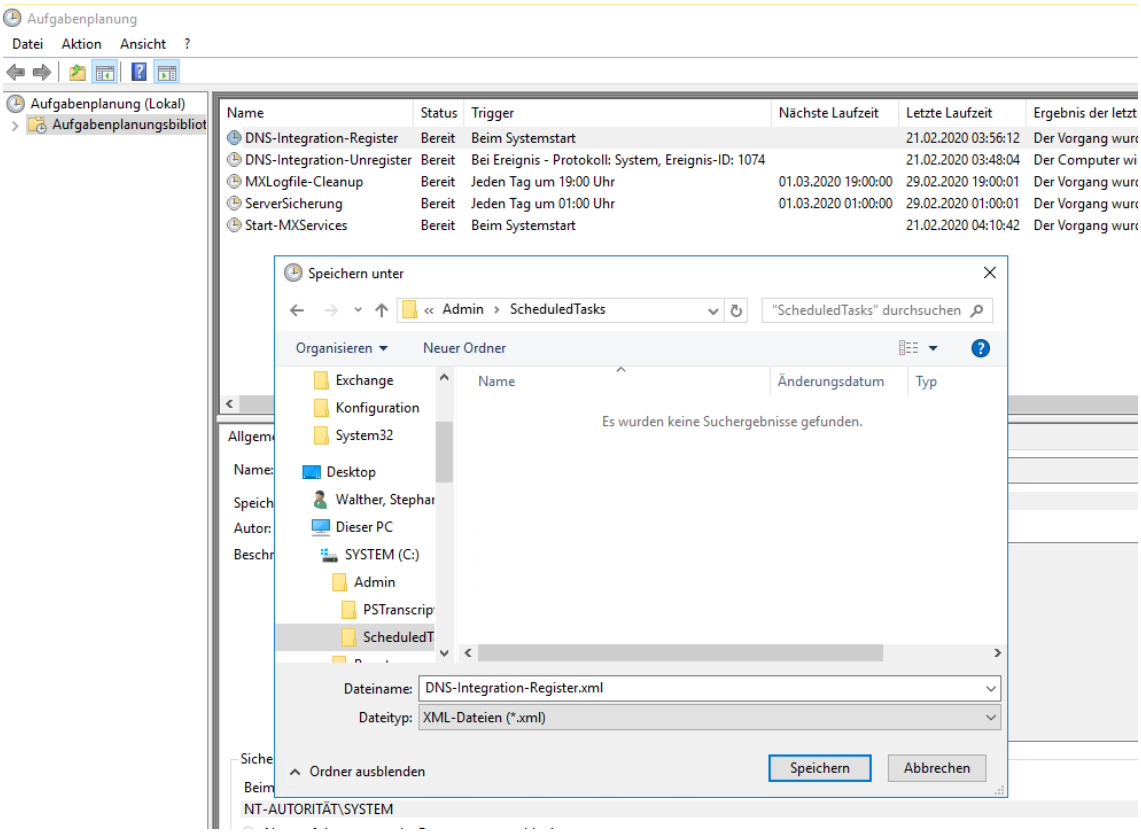


Danach kann ich mich auch schon anmelden. Weiter geht es mit den Windows Updates. Damit der Server an den Windows Update Server im Internet kommt, hänge ich ihn fix in ein freigeschaltetes Netzwerk. Und dann geht es auch schon los:

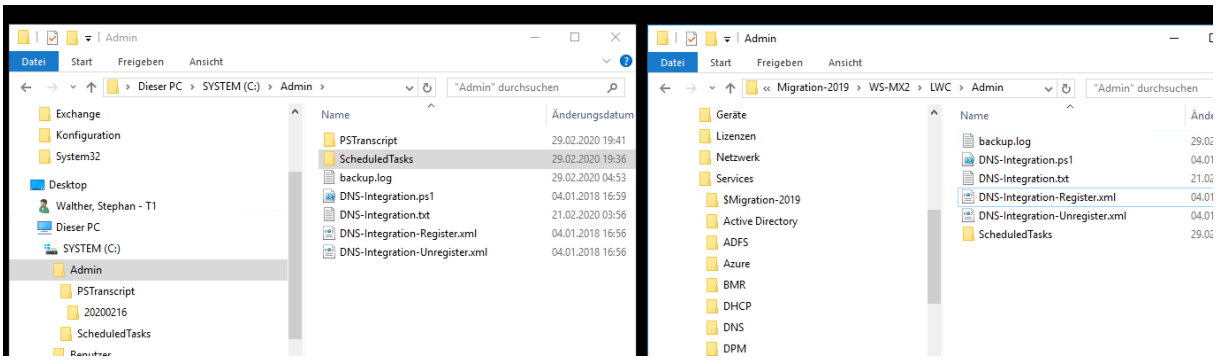


Sammlung von Informationen und Elementen im alten Server

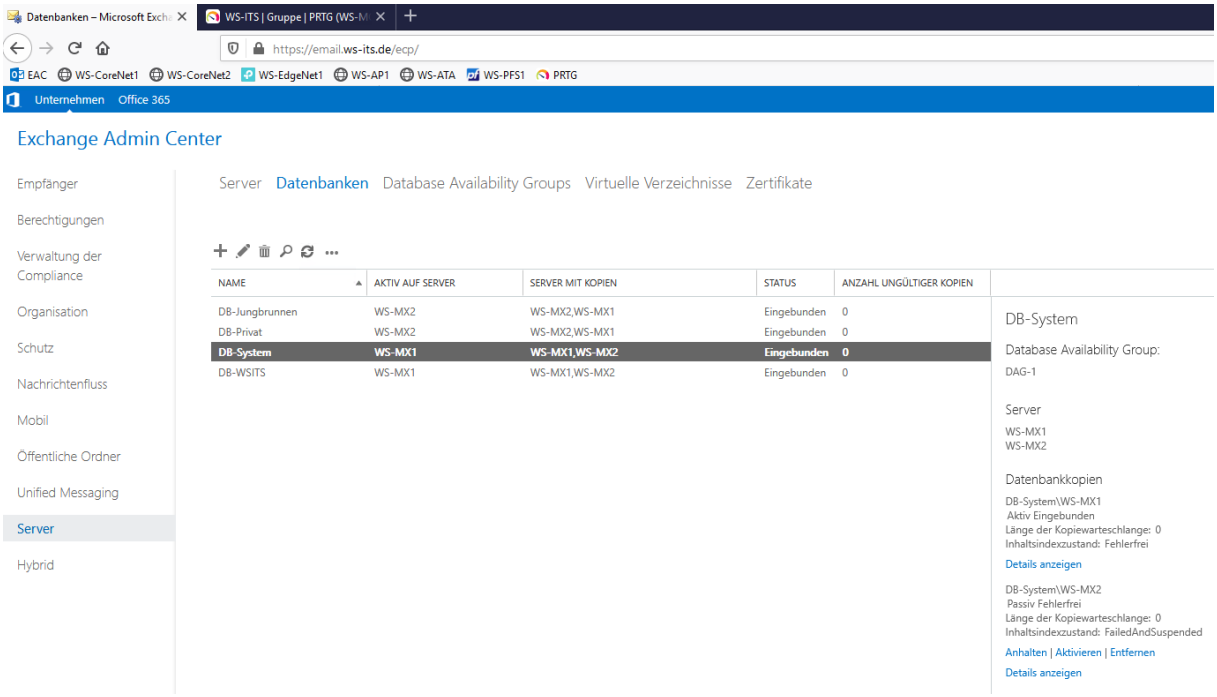
Die Aktualisierung des neuen Servers wird einige Minuten dauern. Währenddessen prüfe ich, ob es auf dem alten WS-MX2 noch brauchbare Dateien und Konfigurationen gibt: Dazu zählen auch geplante Aufgaben. Diese exportiere ich als XML-Dateien:



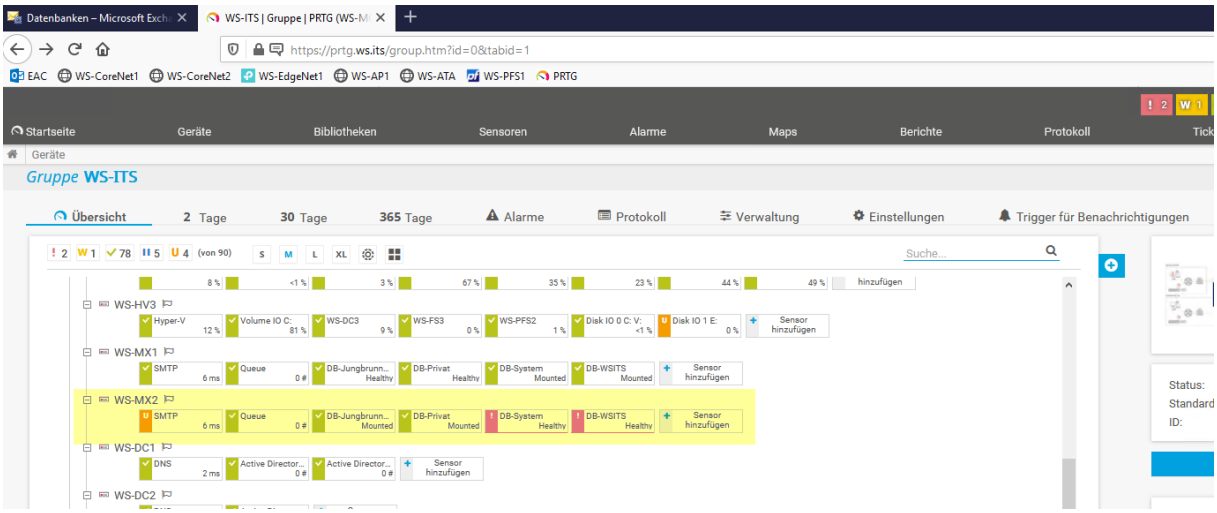
Weitere Dateien lege ich üblicherweise unter C:\Admin ab. Auch diese werden in ein Netzlaufwerk kopiert:



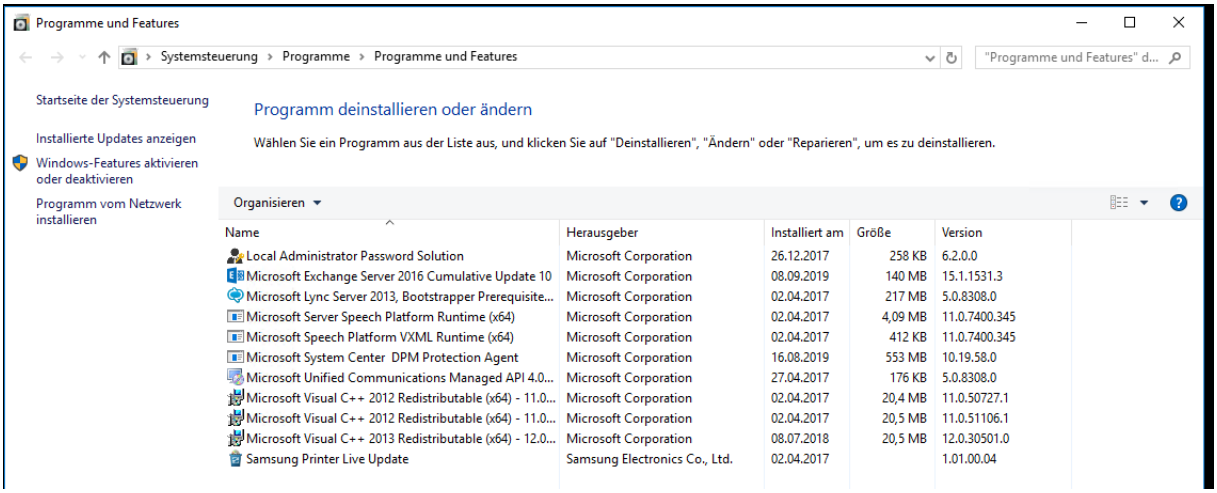
Meine Exchange Server laufen aktuell nicht fehlerfrei. Hier sind meine 4 Datenbanken sichtbar. Die DB-System hat auf dem Server WS-MX2 Probleme mit dem Suchindex. Dadurch kann ich die Datenbank nicht auf WS-MX2 bereitstellen:



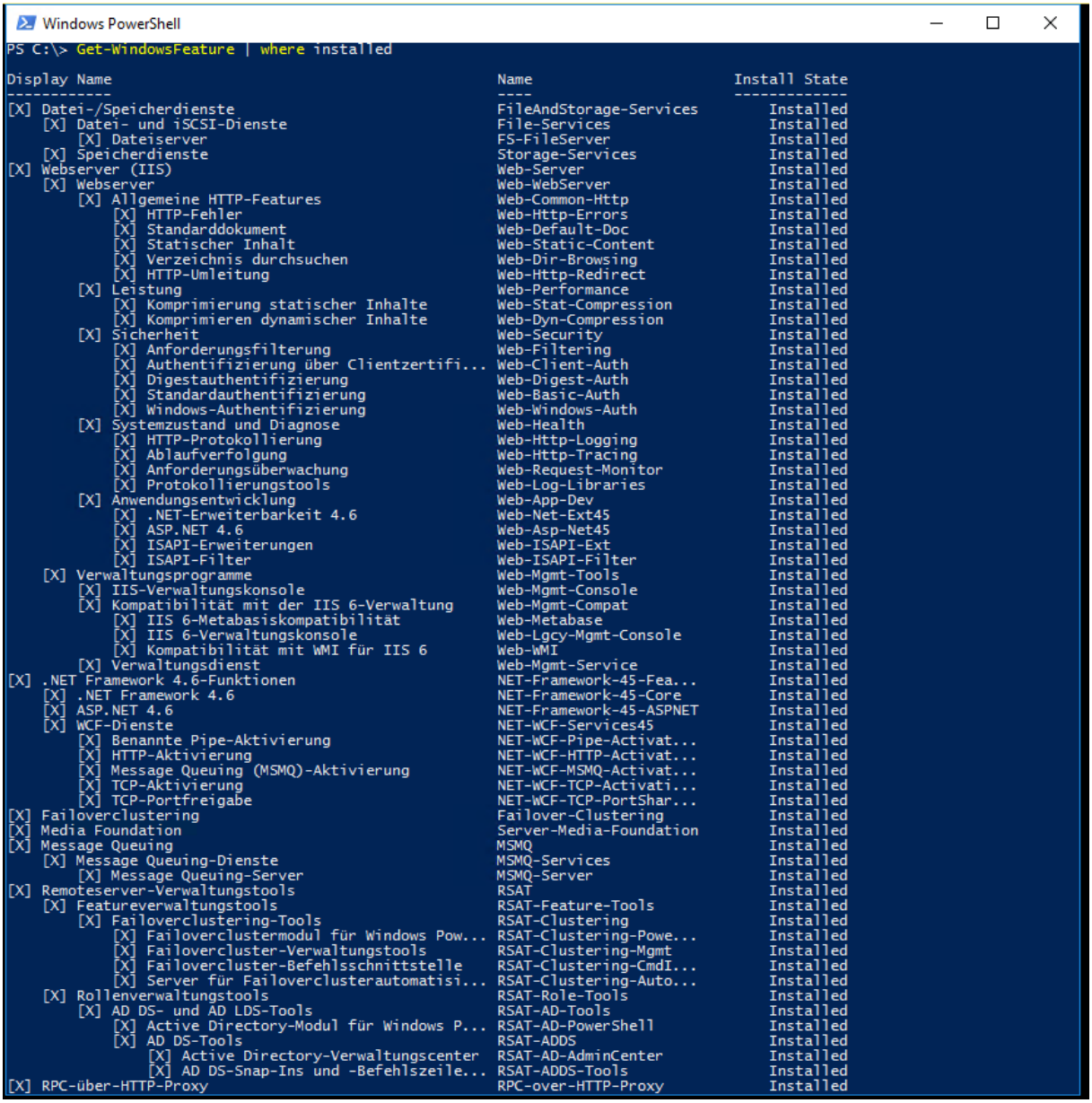
In meinem Monitoring ist das ebenfalls als Problem sichtbar:

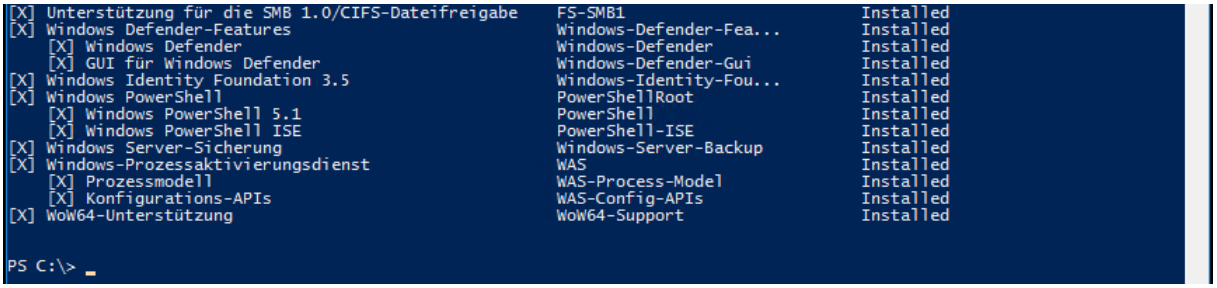


Für die Neuinstallation ist es auch immer interessant, welche Anwendungen auf dem alten Server installiert waren. Viel ist es nicht, da der Server nur für Exchange verwendet wird:

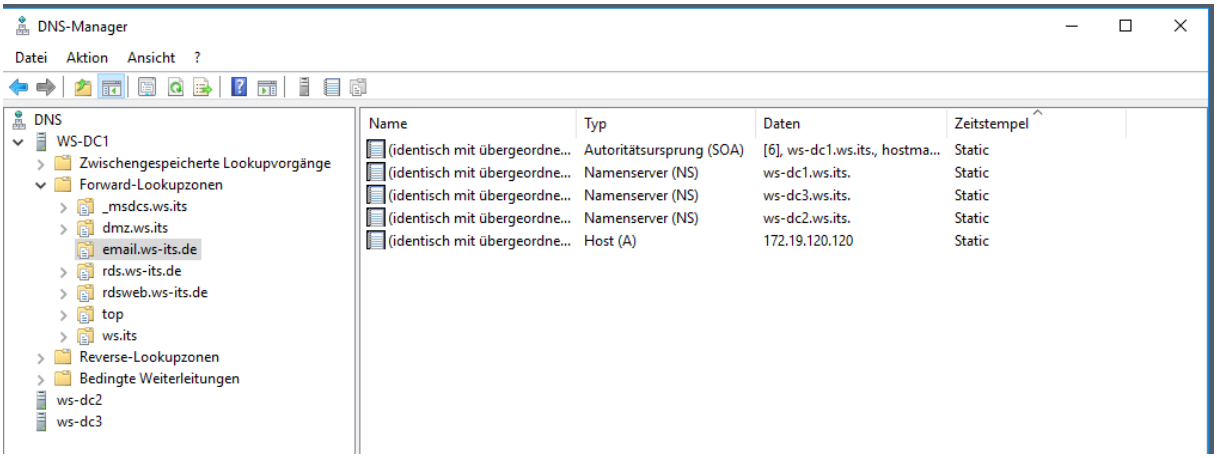


Die installierten Rollen und Features lese ich fix mit der Powershell aus:

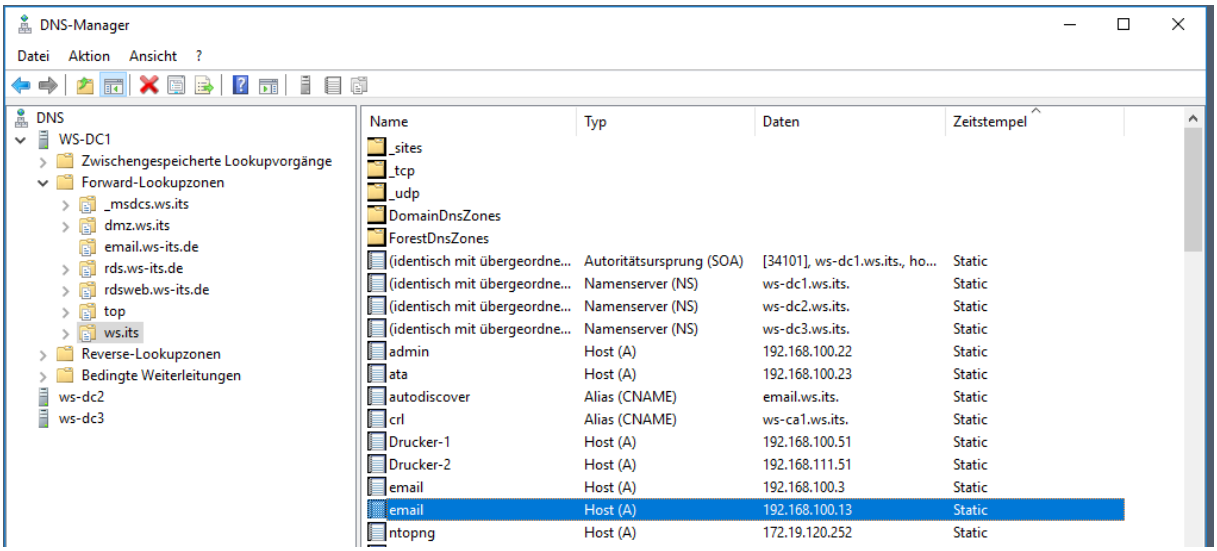




Im DNS ist erkennbar, wie ich für interne Clients die interne Adresse auf den LoadBalancer konfiguriert habe: Es existiert dafür eine eigene DNS-Zone für den externen Namespace:



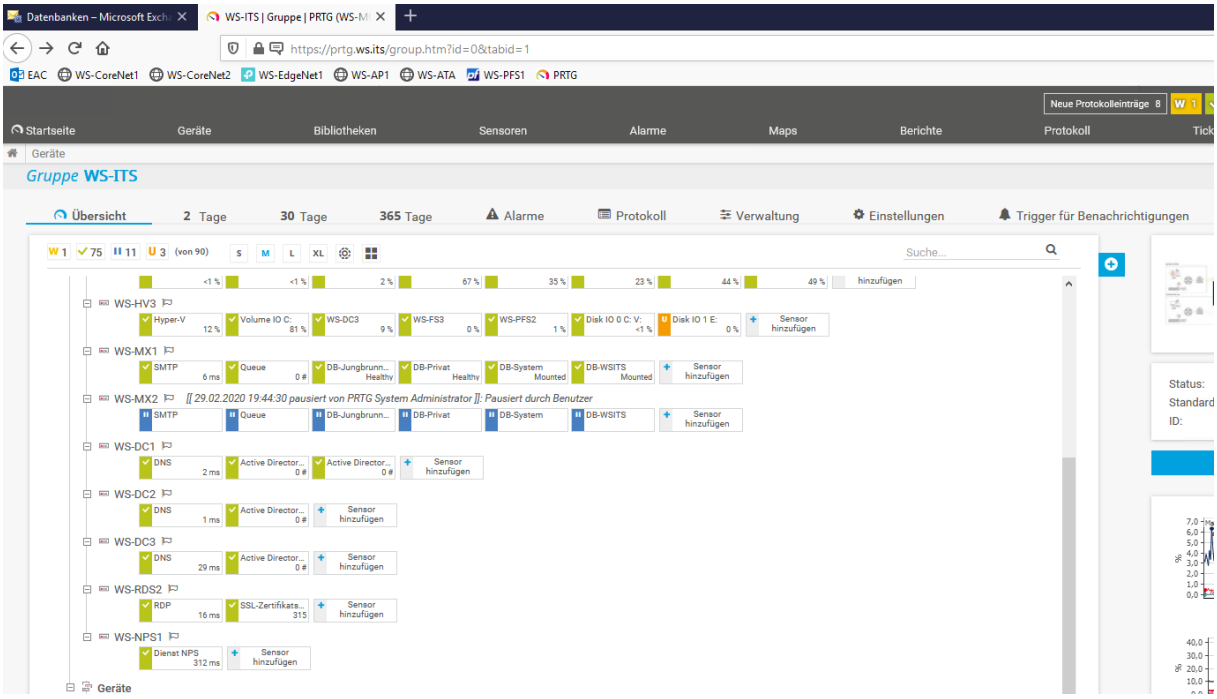
Früher verwendete ich dafür den zusätzlichen Namespace email.ws.its, dessen Konfiguration einfache HOST-A-Records in meiner Domain-Zone sind:



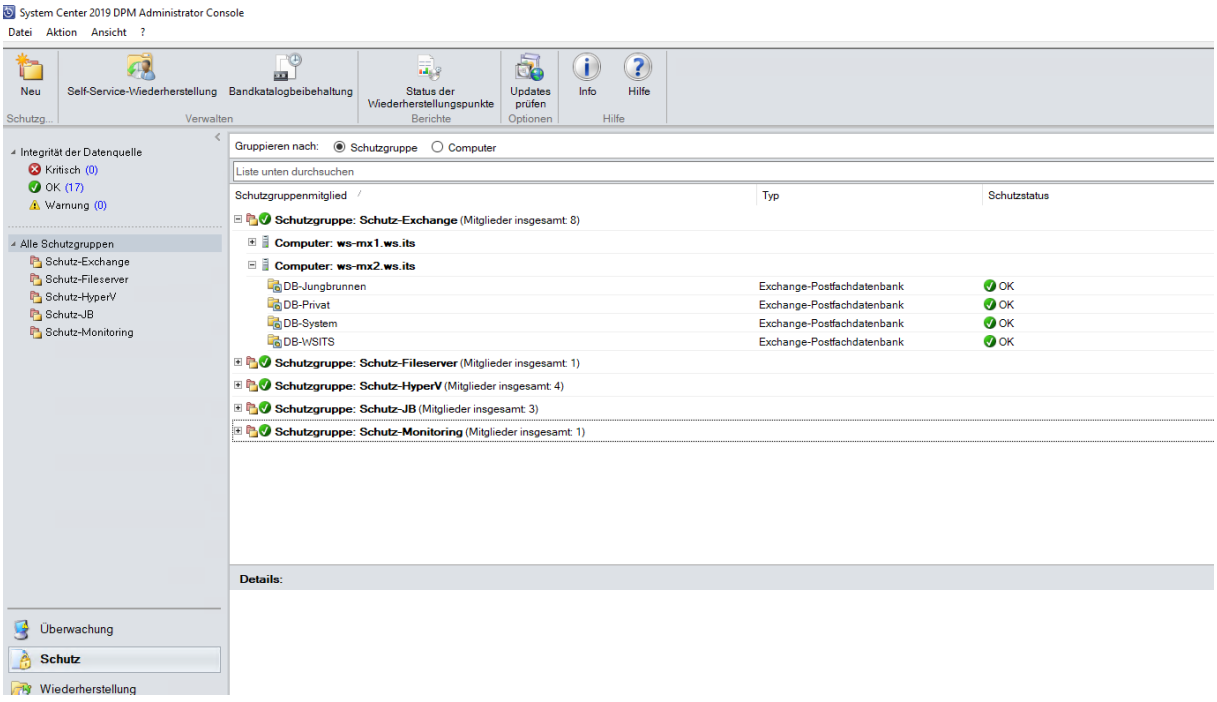
Mehr ist auf meinem Server WS-MX2 nicht zu finden.

Maintenance vorbereiten

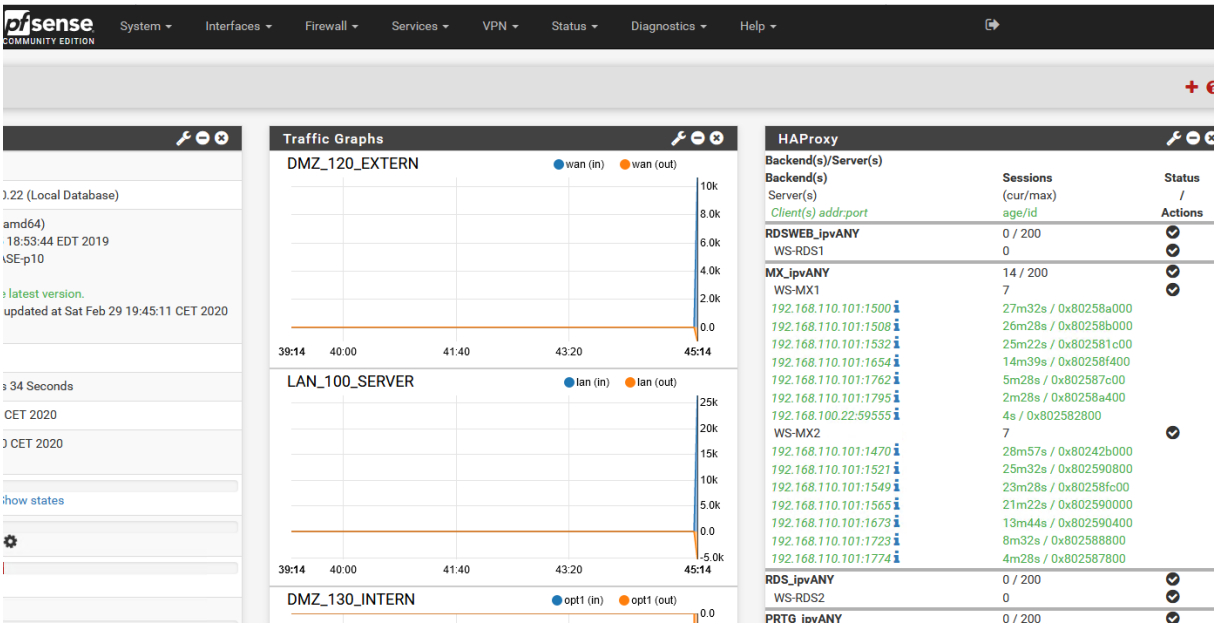
Bevor ich die Deinstallation starte, halte ich die Sensoren des Servers in meinem PRTG-Monitoring an:



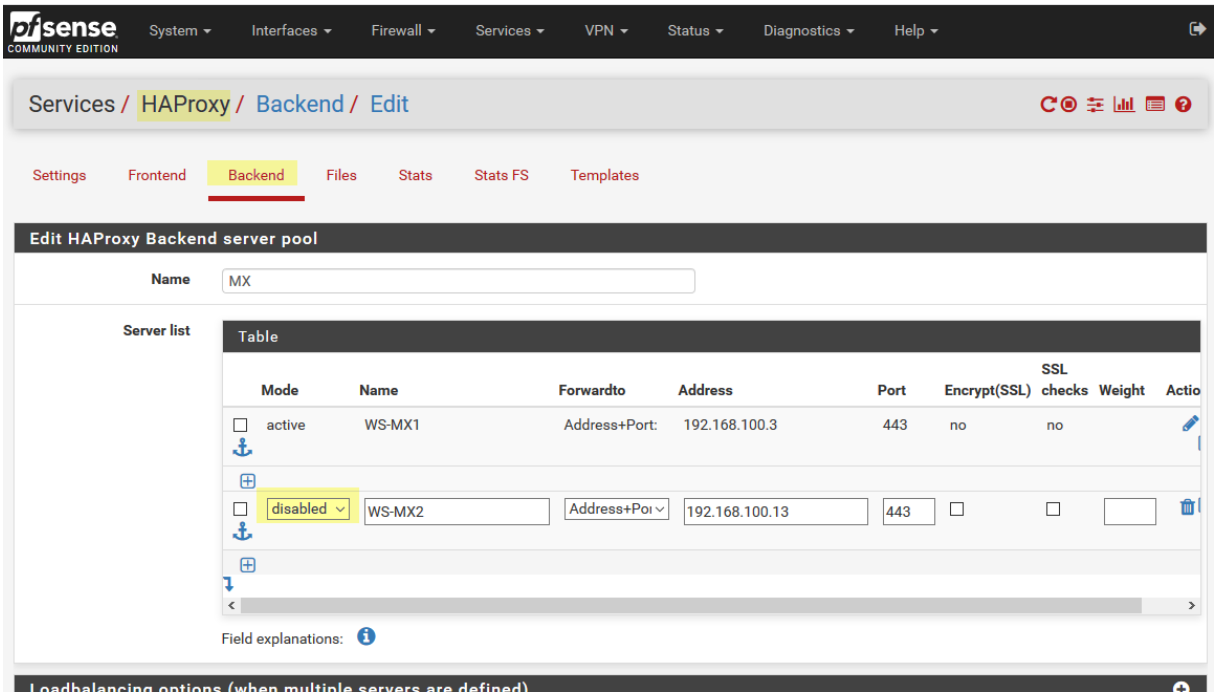
Ebenso kontrolliere ich noch den aktuellen Sicherungsstatus meiner Datenbanken. Diese werden von einem System Center Data Protection Manager 2019 (DPM) gesichert. Hier ist alles gut:



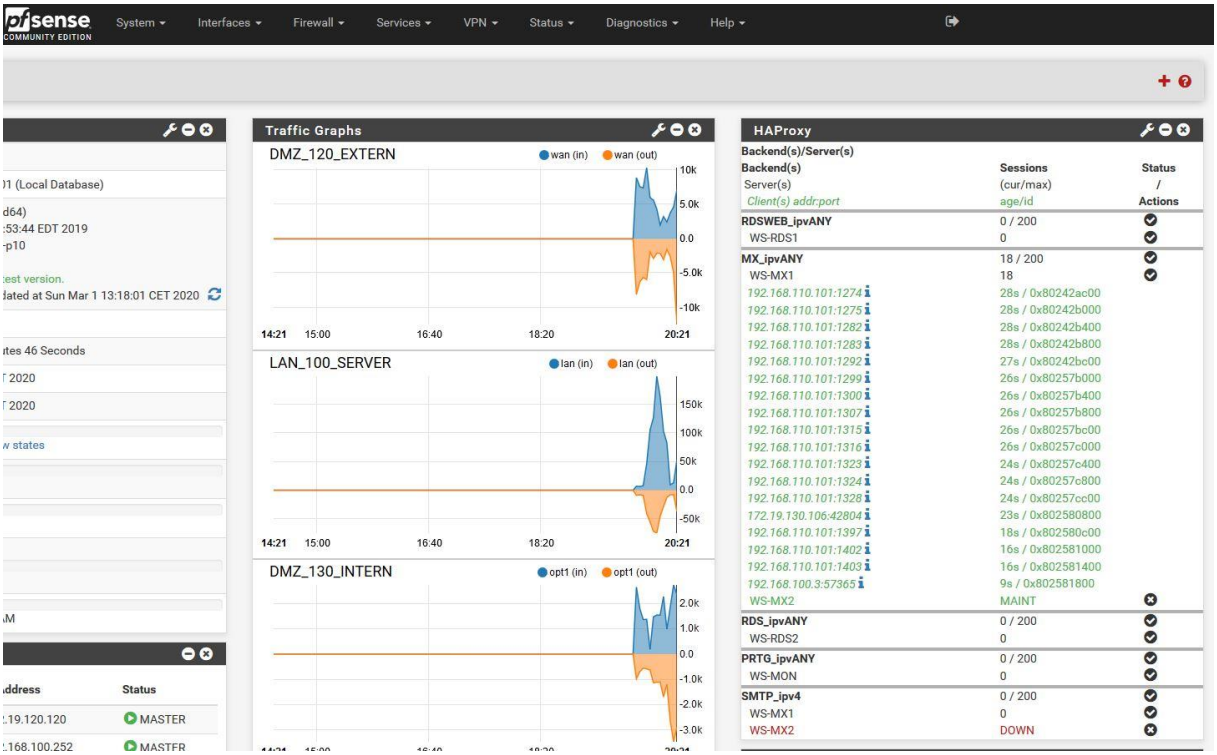
In meiner PFSense starte ich im Modul HAProxy die Maintenance für HTTPS und SMTP für den Server WS-MX2:



Die Maintenance wird über das Backend eingestellt:



Nach wenigen Sekunden schwenken die Verbindungen der Clients auf den Server WS-MX1:



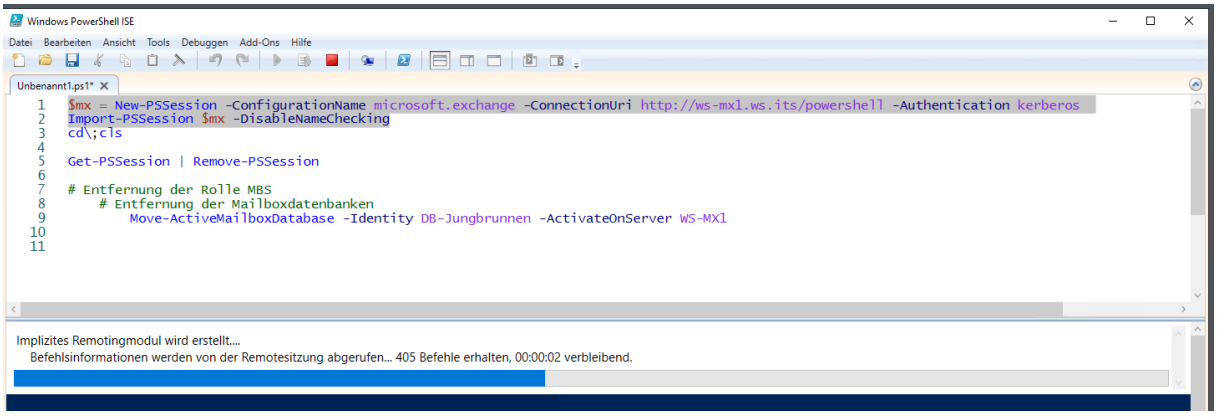
Es kann losgehen.

Entfernung der alten Exchange-Installation

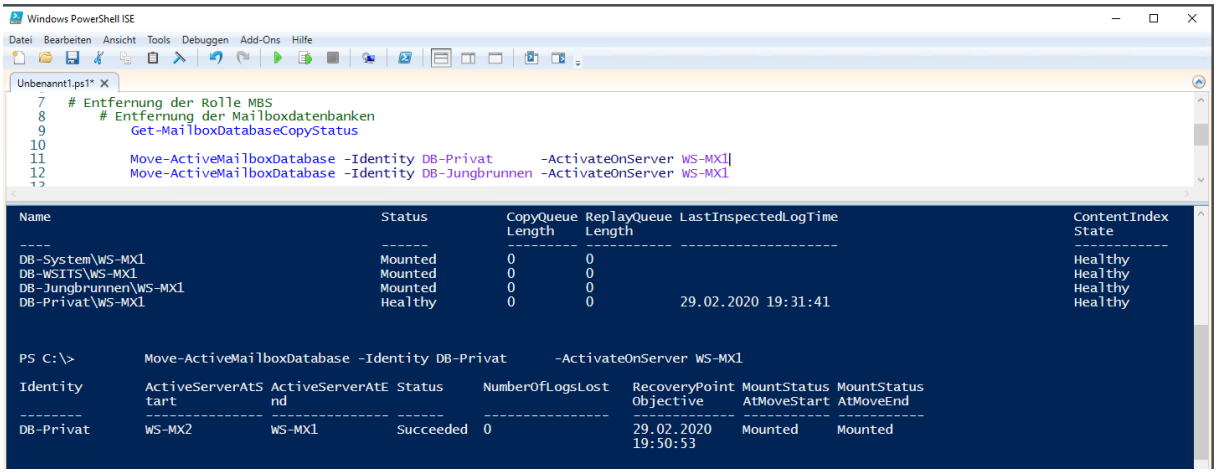
Bereinigungen in der Rolle MBS

Alle Schritte der Migration des Exchange Servers habe ich mit einem PowerShell-Script vorbereitet. In diesem Abschnitt baue ich die Konfigurationen der Rolle Mailboxserver zurück.

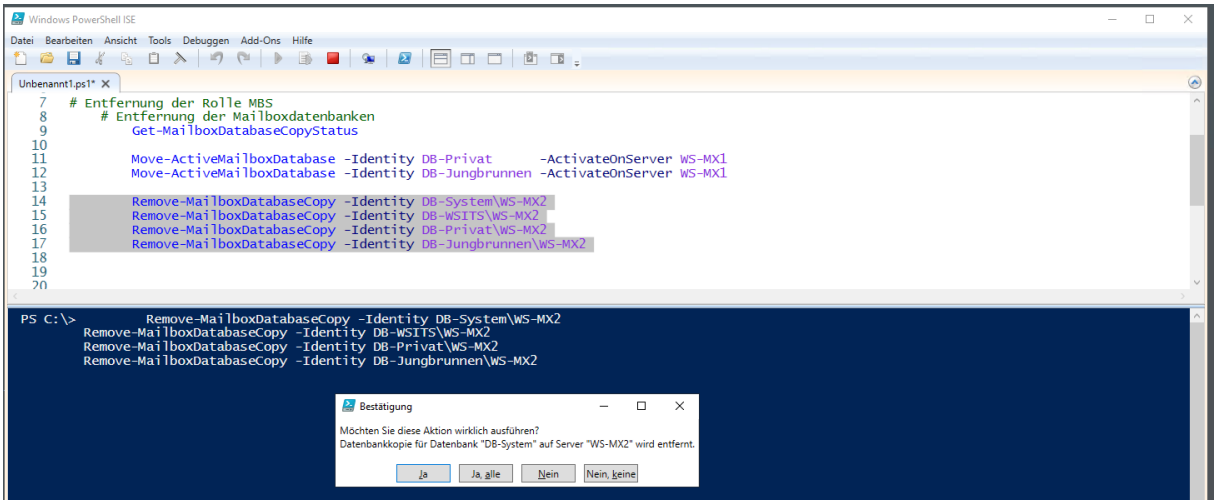
Zuerst verbinde ich meine PowerShell-ISE mit dem Exchange Server:



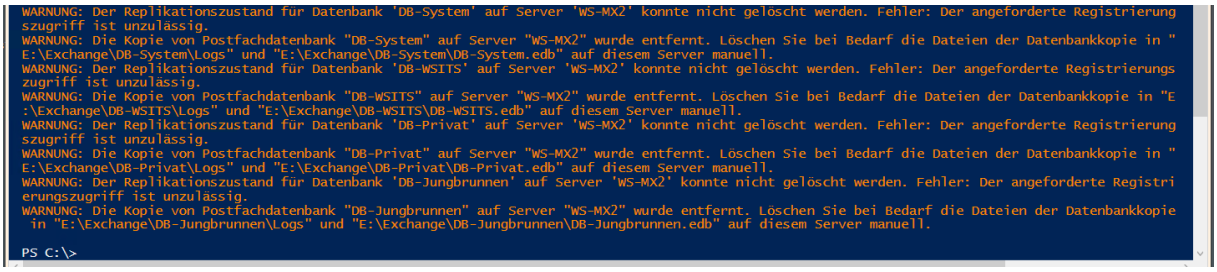
Jetzt verschiebe ich die 2 noch aktiven Datenbanken auf den Server WS-MX1:



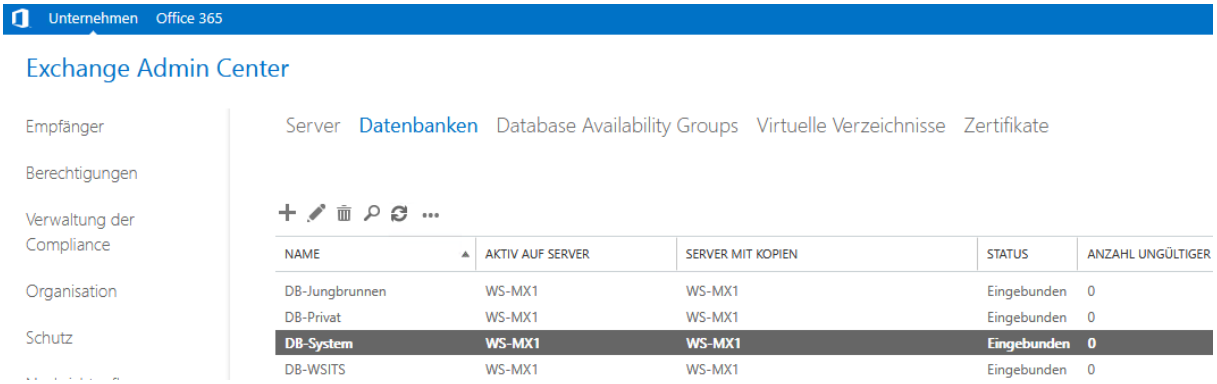
Nachdem nun alle 4 Datenbanken vom Server WS-MX1 bereitgestellt werden kann ich die 4 Datenbankkopien vom Server WS-MX2 entfernen. Jede Aktion wird manuell bestätigt



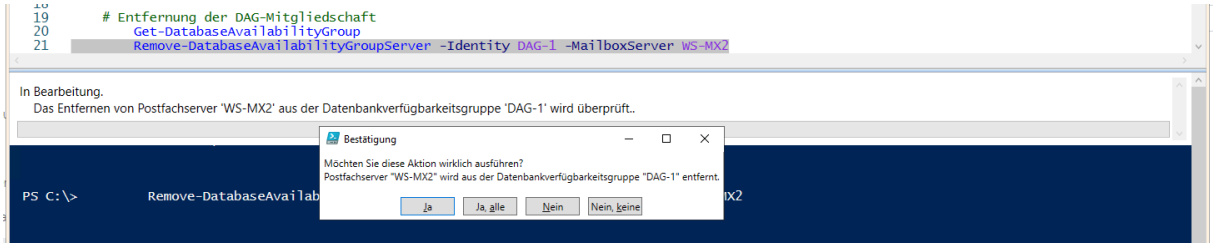
Der Befehl kann die Datenbank-Dateien im Dateisystem nicht löschen. Daher erhalte ich für jede entfernte Kopie eine Warnung. Diese kann ich ignorieren, da auch die virtuelle Festplatte darunter später entfernt wird:



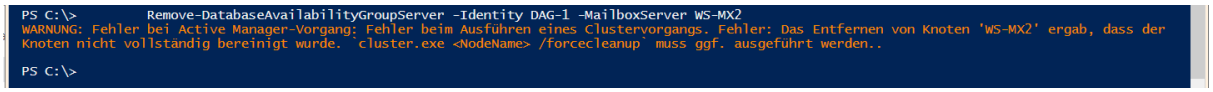
Ein letzter Blick im Exchange Control Panel zeigt, dass alle Datenbanken ausschließlich auf Server WS-MX1 laufen:



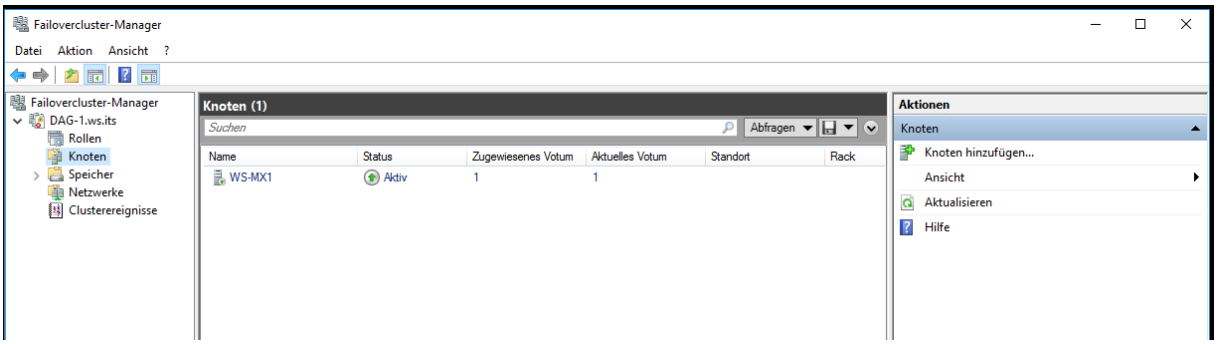
Beide Mailserver laufen in einer Datenbankverfügbarkeitsgruppe. Ich entferne den Server WS-MX2 als Mitglied:



Als Ergebnis wird mir eine Warnung angezeigt. Das ist nicht normal:



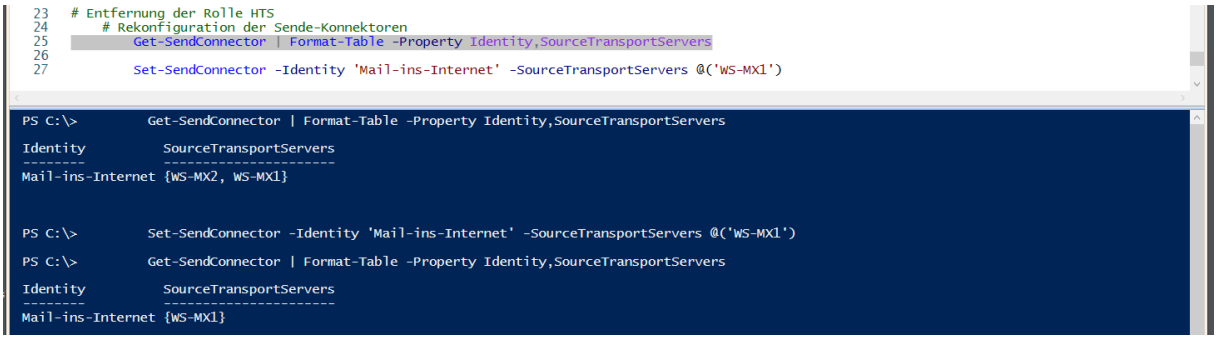
Daher kontrolliere ich mit dem Failovercluster-Manager die Lage. Der Server wird nicht mehr als Clusterknoten aufgeführt. Es sollte also kein Problem sein:



Der Server WS-MX2 ist damit kein Mailboxserver mehr.

Bereinigungen in der Rolle HTS

Weiter geht es mit der Rolle Hub-Transport-Service – also dem Nachrichtenfluss. Beide Server dürfen Mails ins Internet senden. Dafür verwenden sie einen Sende-Konnektor. Mit Set-SendConnector entferne ich den Server WS-MX2. Jetzt darf nur noch Server WS-MX1 Mails versenden:



Die Receive-Konnektoren werden bei der Deinstallation automatisch entfernt, da sie serverbezogen sind. Hier sichere ich nur die aktuelle Konfiguration in einer Textdatei:

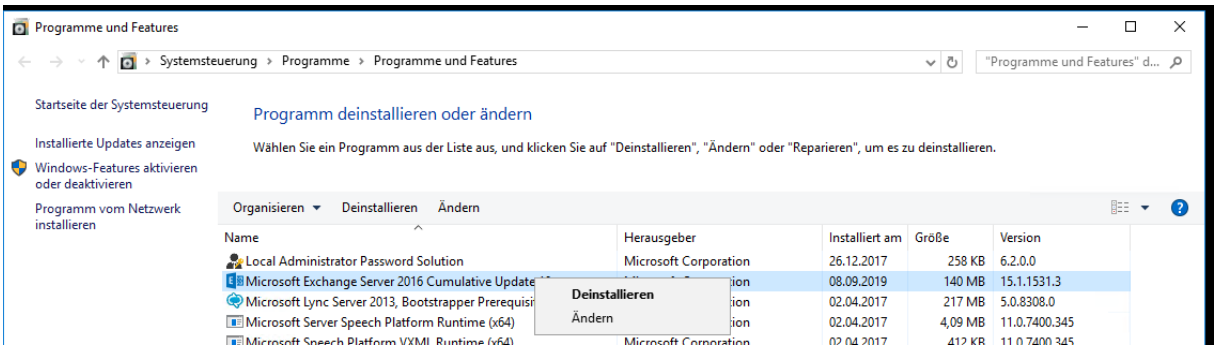
```
29      # Sichtung der Receive-Konnektoren
30          Get-ReceiveConnector -Server WS-MX2 |
31              Format-List -Property * |
32                  Out-File -FilePath M:\AdminArea\Services\Exchange\Migration-2019\WS-MX2\Receive-Konnektoren.txt
```

```
PS C:\> Get-ReceiveConnector -Server WS-MX2 |
            Format-List -Property * |
                Out-File -FilePath M:\AdminArea\Services\Exchange\Migration-2019\WS-MX2\Receive-Konnektoren.txt

PS C:\> Get-ReceiveConnector -Server WS-MX2

Identity                                Bindings                Enabled
--------                                --------                -------
WS-MX2\Client Proxy WS-MX2              {[::]:465, 0.0.0.0:465} True
WS-MX2\Default WS-MX2                   {0.0.0.0:2525, [::]:2525} True
WS-MX2\Outbound Proxy Frontend WS-MX2   {[::]:717, 0.0.0.0:717} True
WS-MX2\Client Frontend WS-MX2           {[::]:587, 0.0.0.0:587} True
WS-MX2\Mails-vom-Internet               {0.0.0.0:25}            True
WS-MX2\Default Frontend WS-MX2          {[::]:25, 0.0.0.0:25}   True
WS-MX2\ProbeMails                       {0.0.0.0:25}            True
```

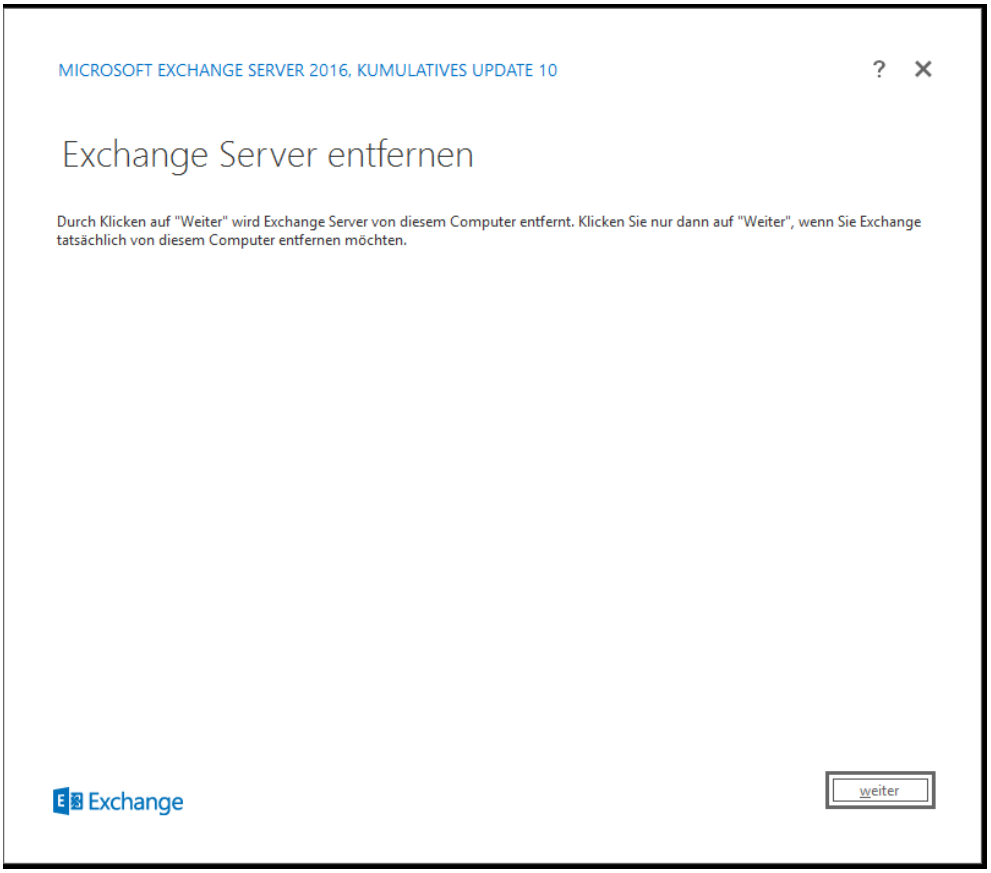
Damit ist auch die Rolle HTS für die Deinstallation vorbereitet. Die Client-Access-Rolle (CAS) ist ebenfalls serverbezogen und kann durch eine einfache Deinstallation des Exchange Servers entfernt werden.

Deinstallation des Exchange Servers – Problem: Reste des Failover Clusters

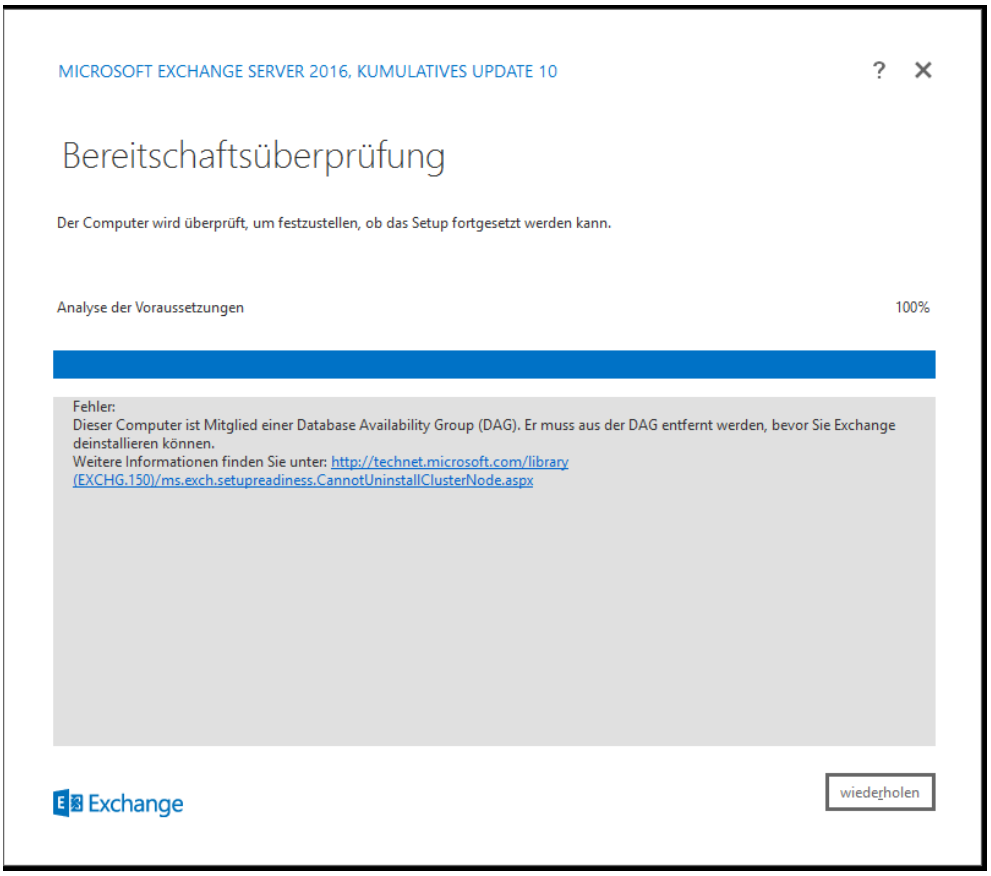
Es kann also mit der Entfernung der Installation weitergehen. Hier hilft die alte Systemsteuerung:



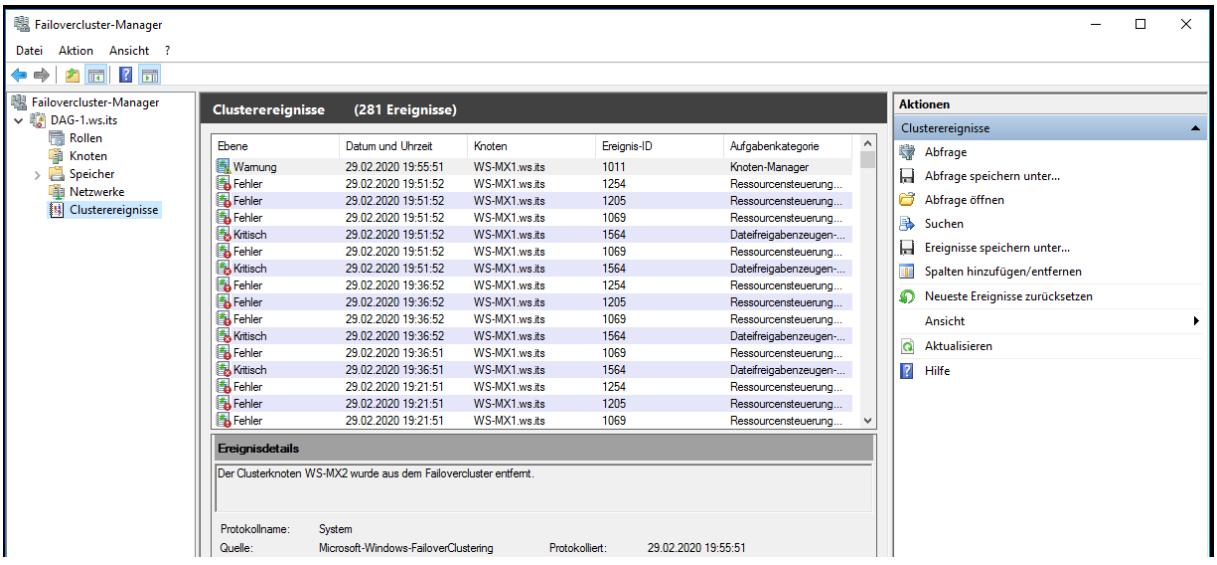
Der Assistent startet:



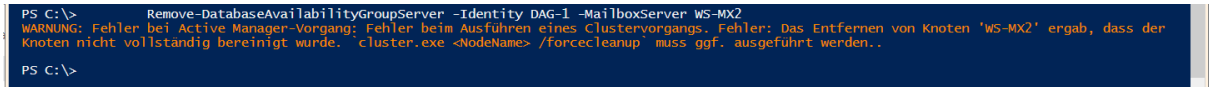
Vor der Deinstallation wird eine Bereitschaftsprüfung ausgeführt. Diese zeigt aber einen seltsamen Fehler an:



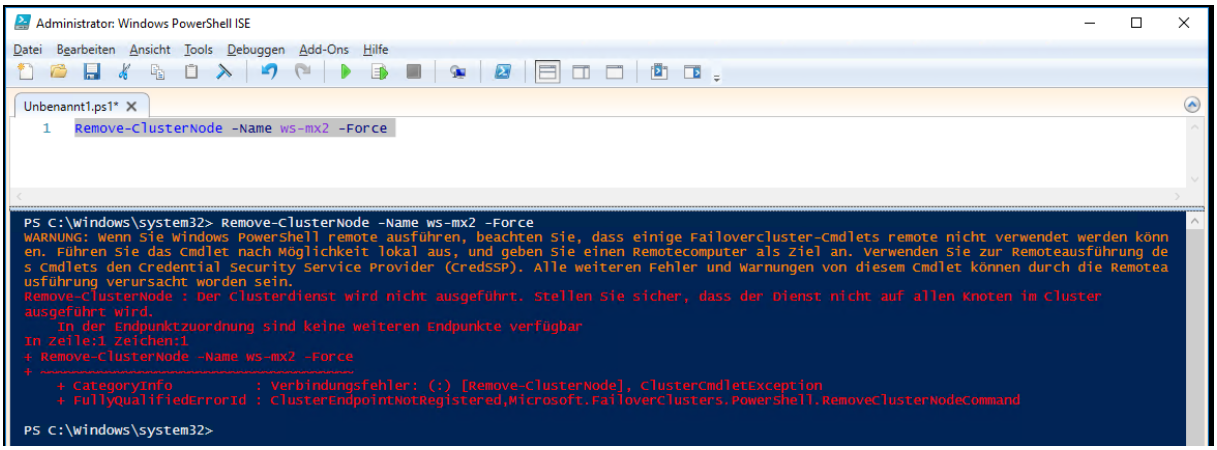
Offensichtlich hat die Entfernung der DAG-Mitgliedschaft (diese wurde ja mit einer Warnung abgeschlossen) nicht funktioniert. Ich öffne erneut den Failovercluster-Manager und prüfe die Cluster-Events. Hier wurde der Server ausgetragen:



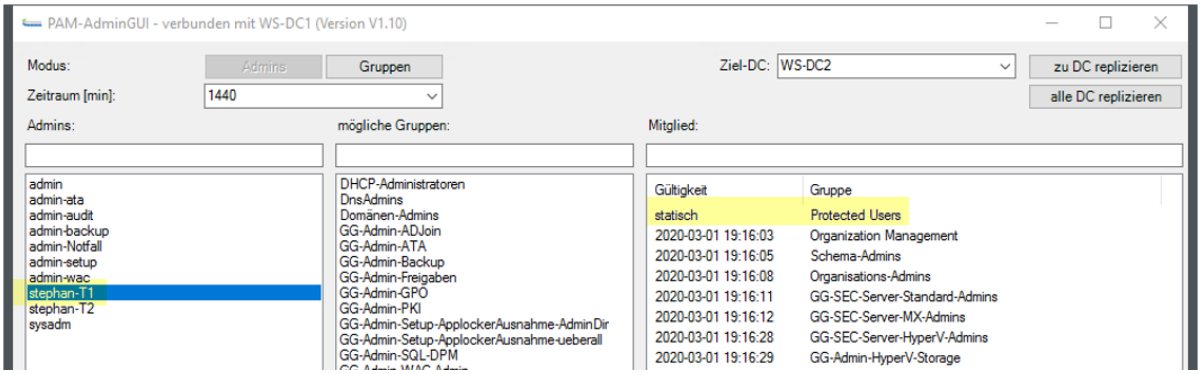
Vielleicht hilft ein Neustart? Normalerweise ist das nicht erforderlich, aber ich versuche es einfach mal. Doch auch der zweite Versuch der Deinstallation scheitert. Dank meines Screenshots von vorhin kann ich mir die Warnmeldung noch einmal genauer ansehen:



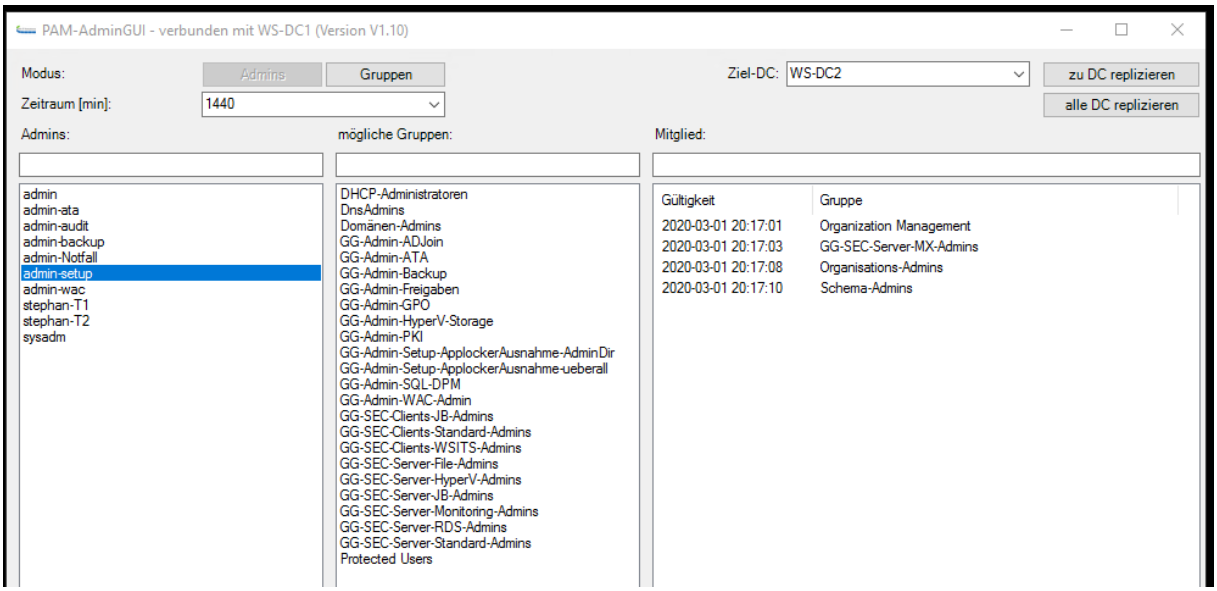
Es gibt eine Nacharbeit. Cluster.exe existiert aber nicht mehr. Daher verwende ich das PowerShell-Cmdlet für die Bereinigung. Leider hat auch dieser Befehl nur eine Fehlermeldung als Ergebnis. Das Cmdlet benötigt einen Kontakt zum Clusterdienst. Doch dieser wurde bereits beendet:



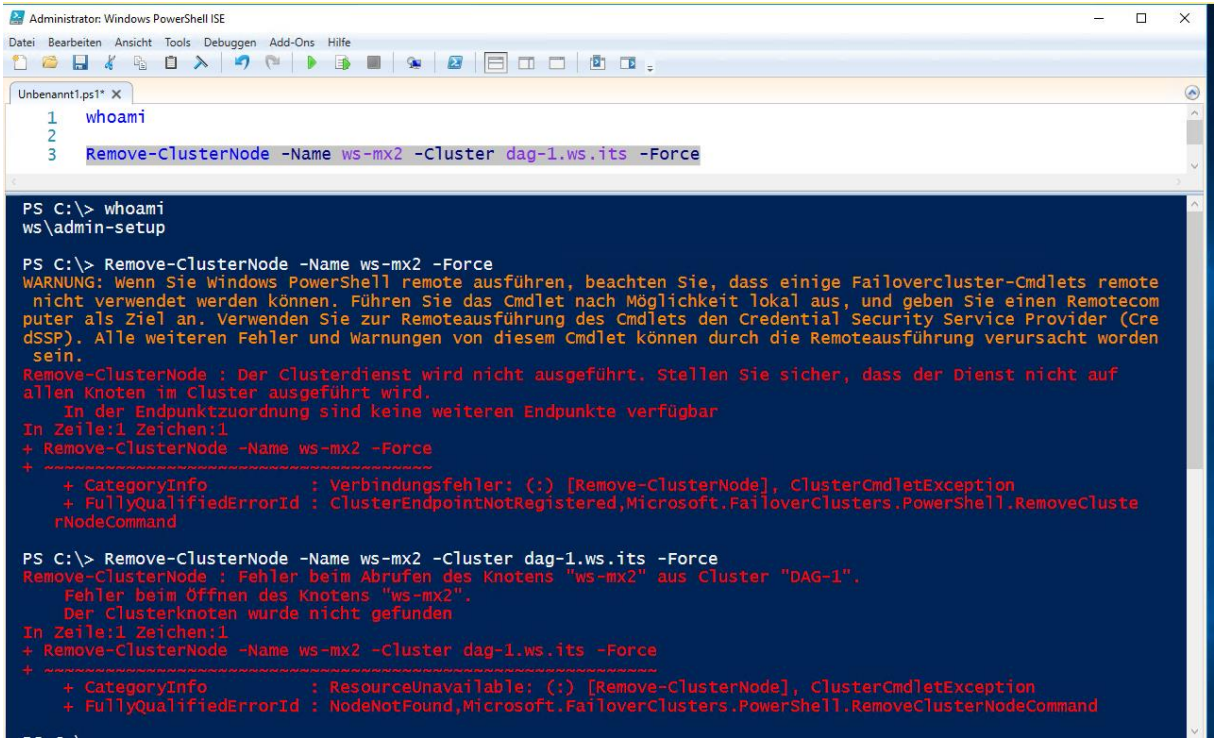
Und dann kommt mir eine Idee: Bis Windows Server 2019 konnte für die Authentifizierung in einem Cluster ausschließlich NTLM verwendet werden. Doch genau dieses alte Protokoll wird explizit für Mitglieder der AD-Gruppe „Protected Users“ abgeschaltet. Und genau in dieser Gruppe ist meine administrative T1-Kennung dauerhaftes Mitglied:



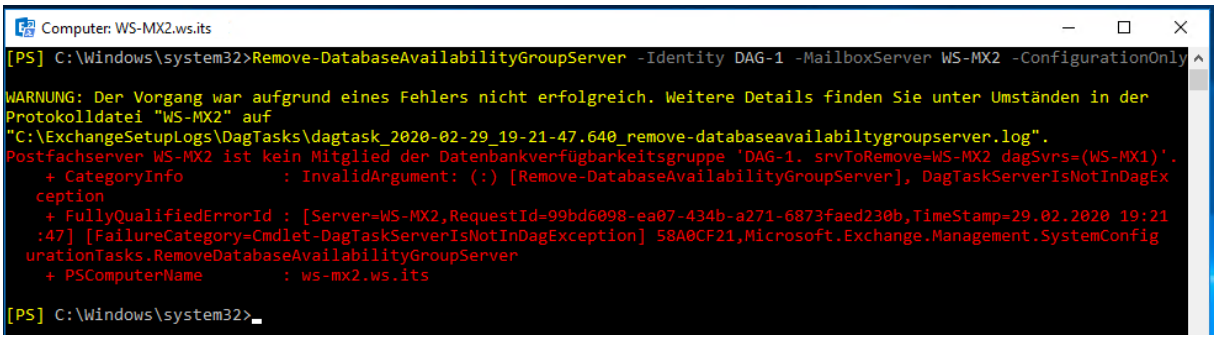
Also präpariere ich einen „ungeschützten“ Account mit den sonst erforderlichen Rechten:



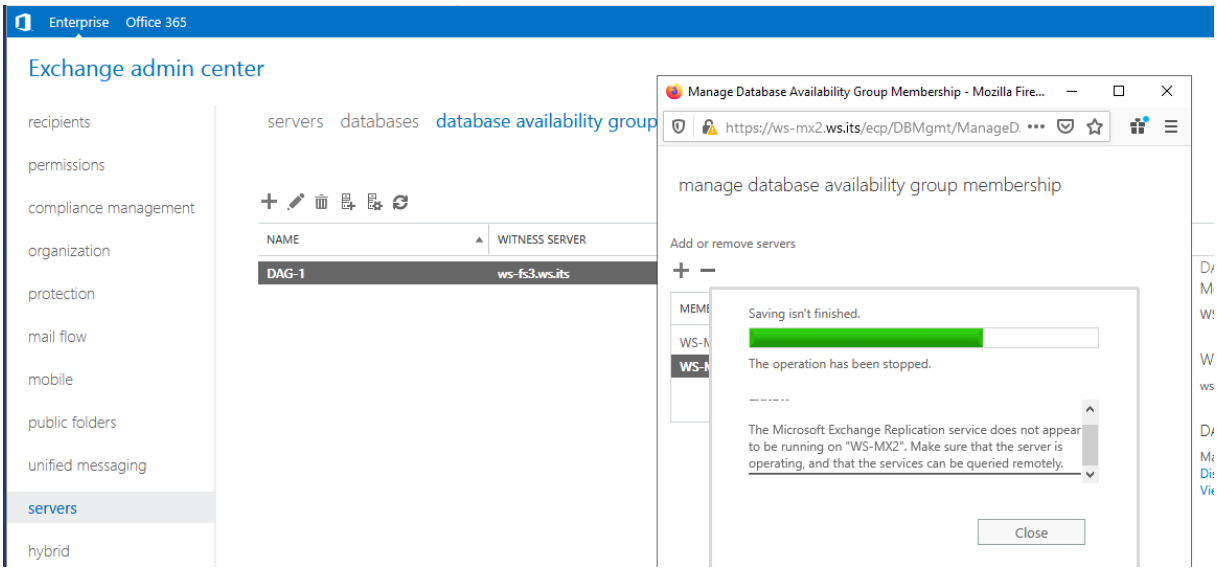
Mit dieser Kennung starte ich eine PowerShell-ISE und versuche die Entfernung:



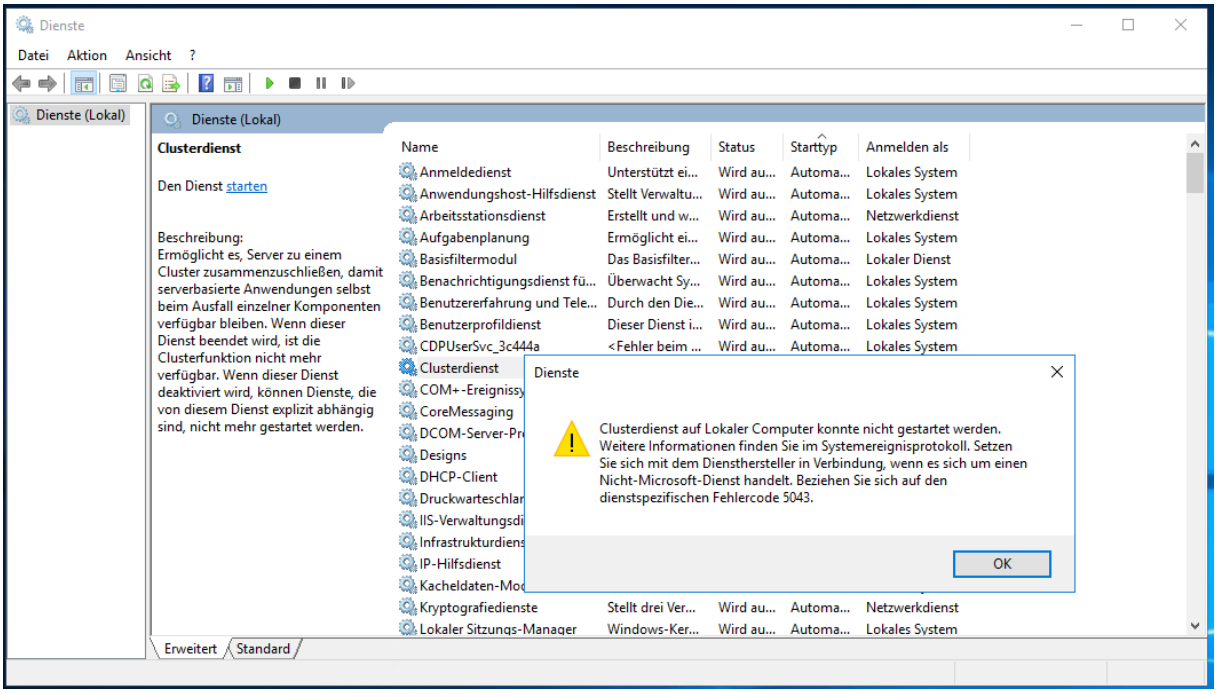
Aber auch hier muss der Cluster-Dienst gestartet sein. Also versuche ich die DAG-Mitgliedschaftsentfernung mit der Kennung meines Admin-Setup. Nur dieser Teil hatte ja schon funktioniert:



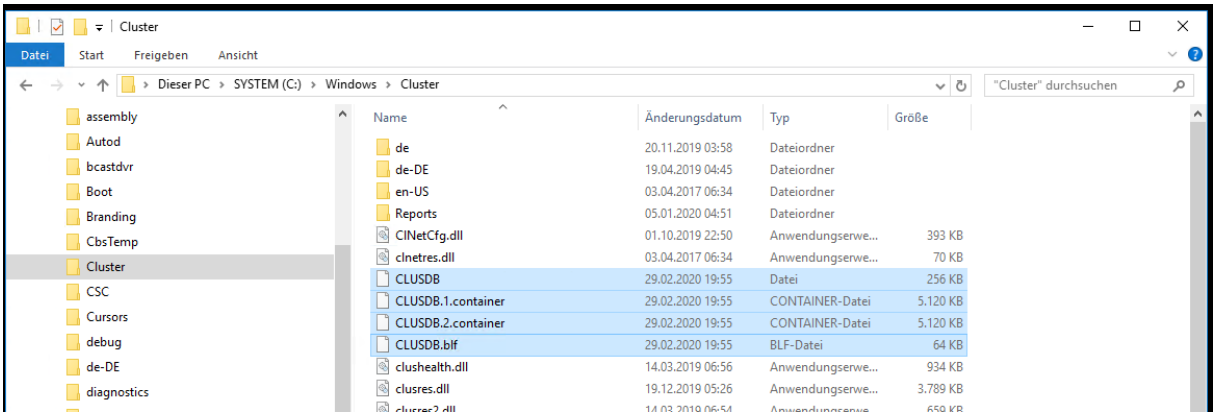
So macht das keinen Spass. Ich prüfe meine Möglichkeiten in der Webseite des Exchange Admin Centers. Hier wird der Server WS-MX2 sogar noch als Member der DAG gelistet. Daher versuche ich hier eine Entfernung. Die Fehlermeldung ist anders – das Ergebnis bleibt gleich:



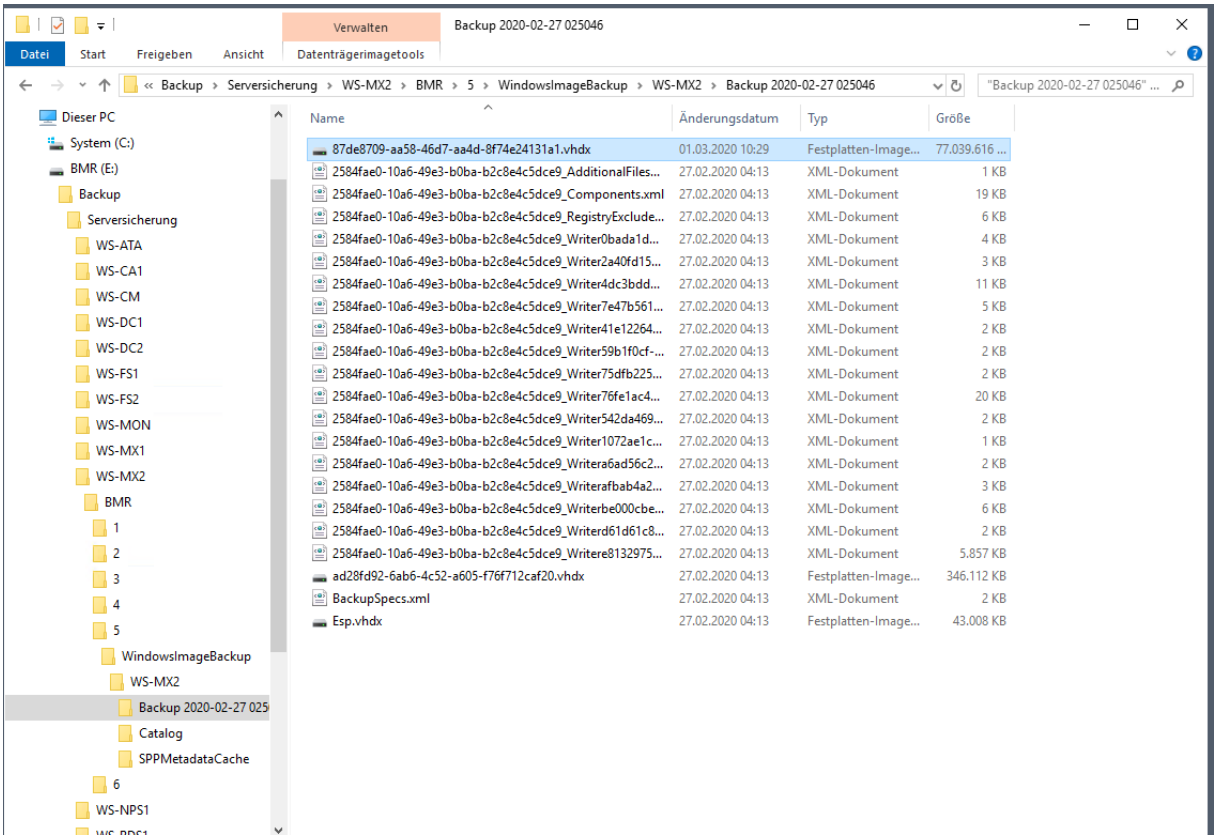
Der Cluster-Dienst selber wird sich nicht mehr starten lassen. Ich versuche es trotzdem einmal. Das wird aber leider auch nichts:



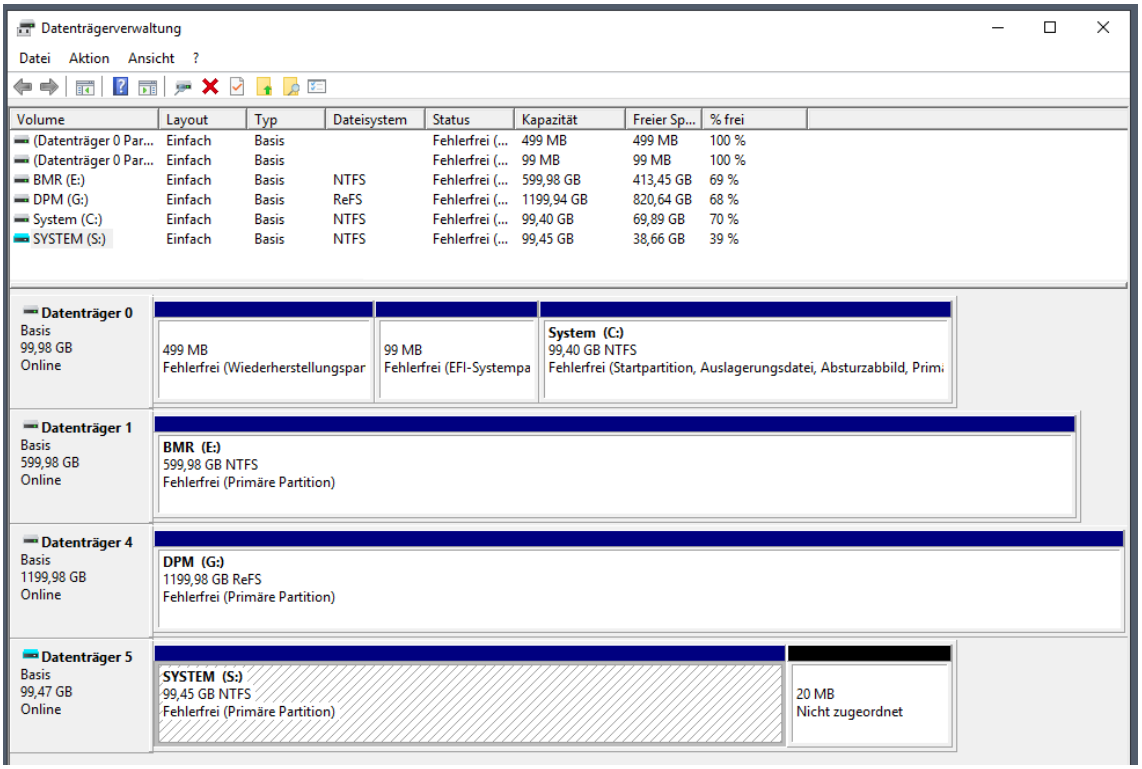
Langsam gehen mir die Möglichkeiten aus. Es wird Zeit für einen Rollback. Der Cluster speichert seine Konfiguration in dem Verzeichnis c:\Windows\Cluster:



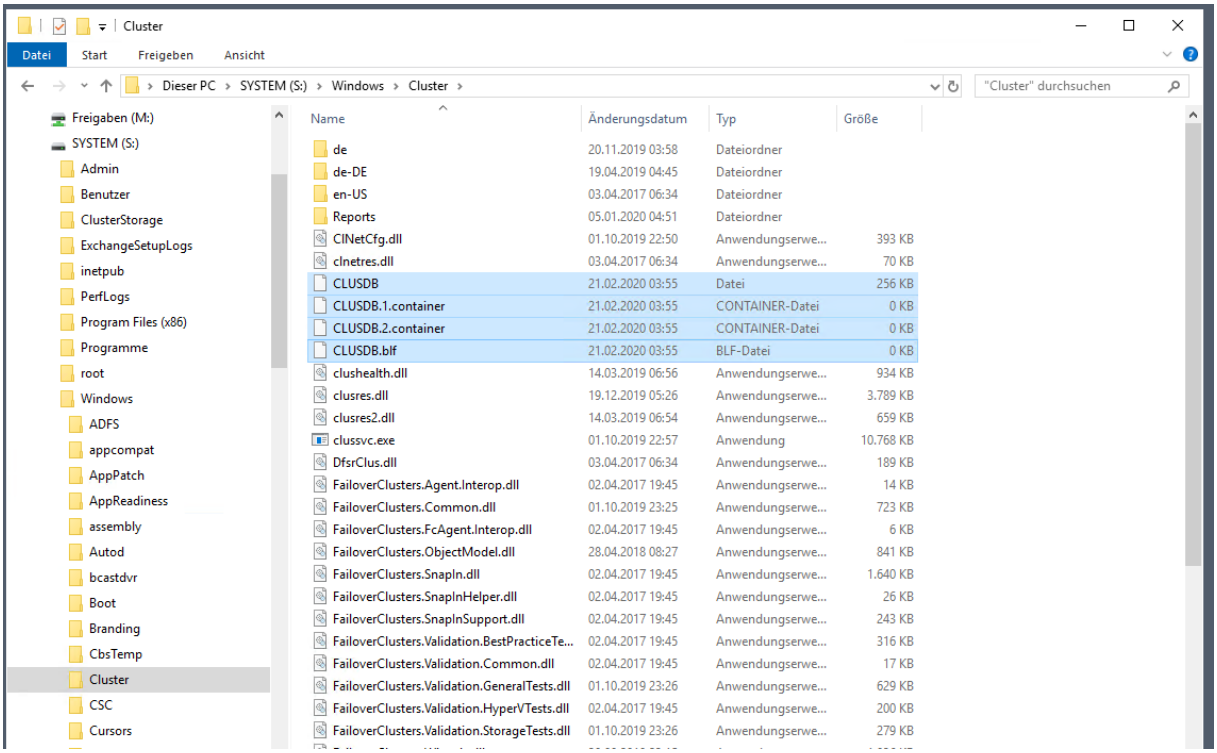
Diese Dateien stelle ich aus einer SystemState-Sicherung wieder her. Jeder Server sichert bei mir seine Betriebssystem-Partitionen mit der Windows Server Sicherung auf ein Netzlaufwerk. Hier liegen also die Sicherungsdateien. Dies sind normale VHDX-Dateien, die sich mit dem Windows Explorer bereitstellen lassen. Ich mounte die passende VHDX auf meinem Sicherungsserver:



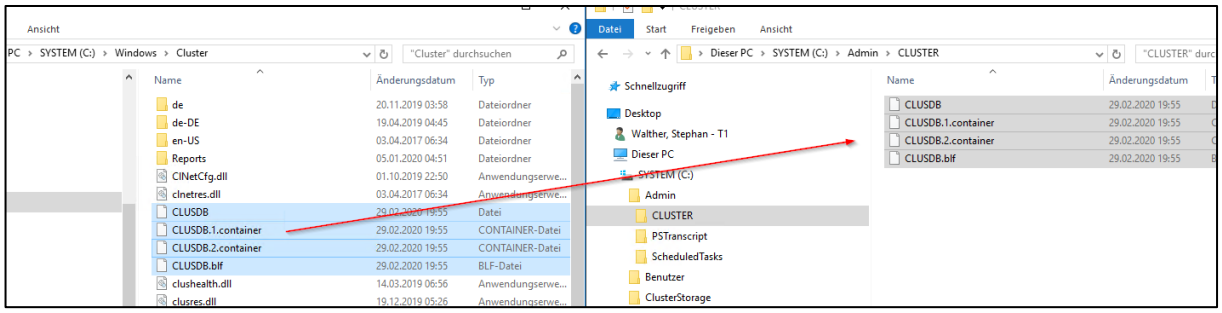
Ggf. müssen die Laufwerksbuchstaben angepasst werden. Hier hilft die Datenträgerverwaltung:



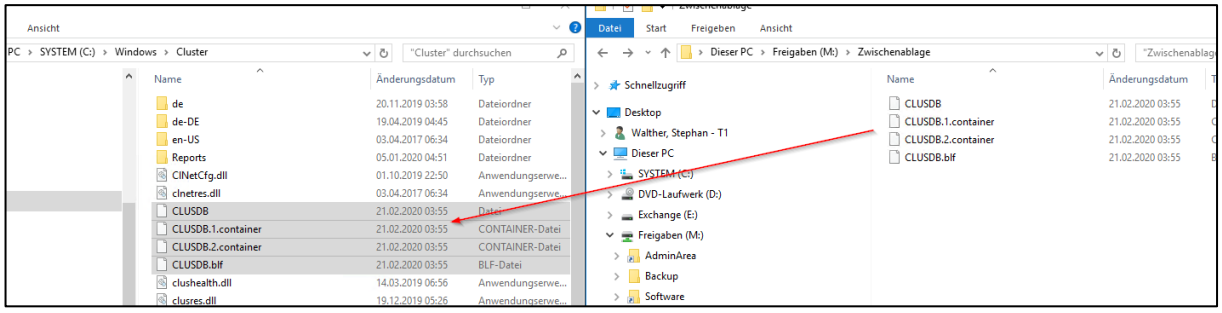
Die Dateien unter S:\. sind wieder von heute Morgen:



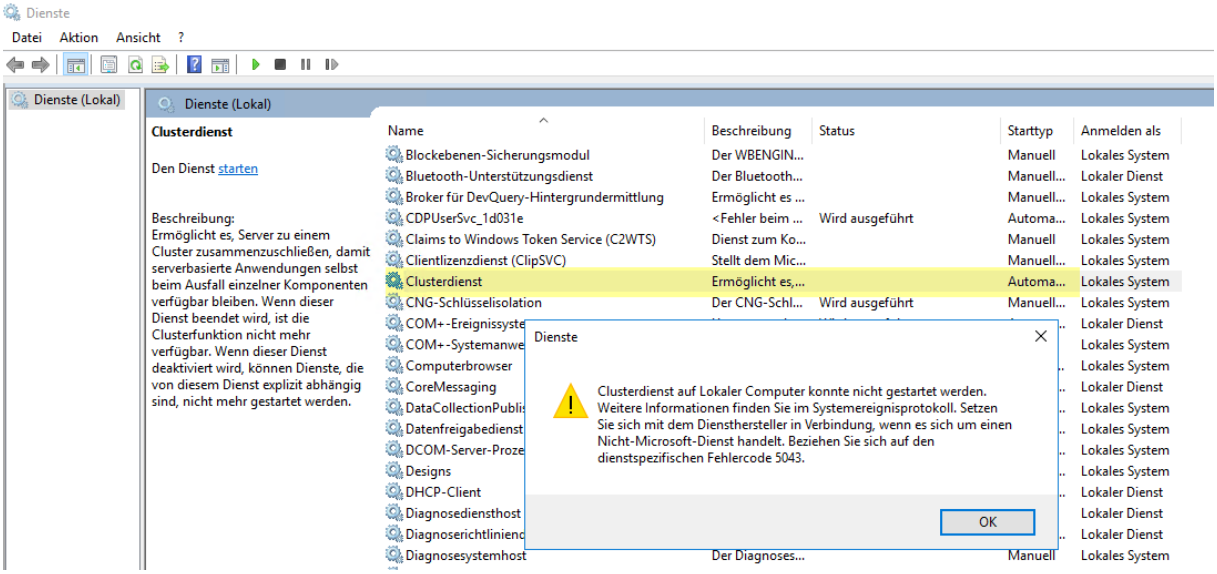
Vor der Wiederherstellung sichere ich noch die aktuelle Datenbank in ein lokales Verzeichnis:



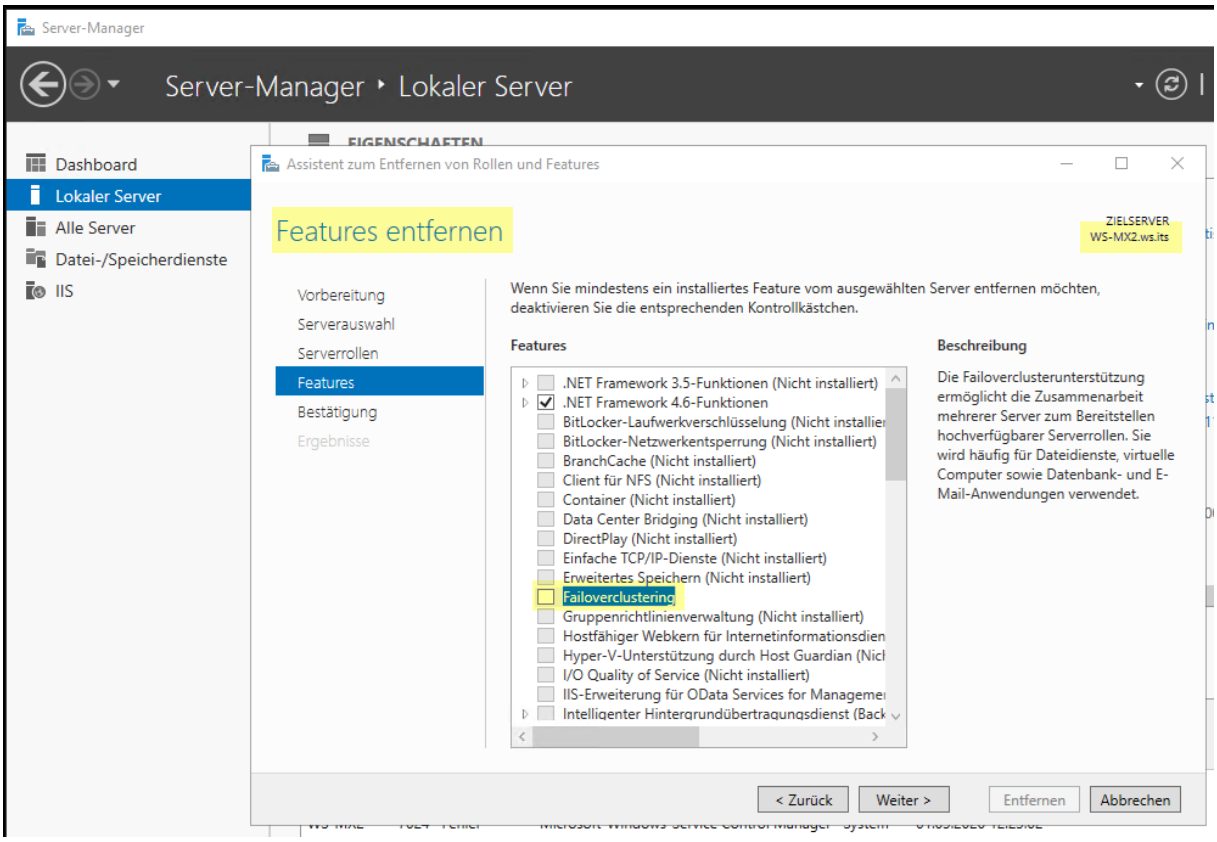
Dann stelle ich die Dateien wieder her. Da mein Exchange Server keinen direkten Zugriff auf meinen Sicherungsserver hat, stelle ich die Dateien über ein Austausch-Netzlaufwerk bereit:



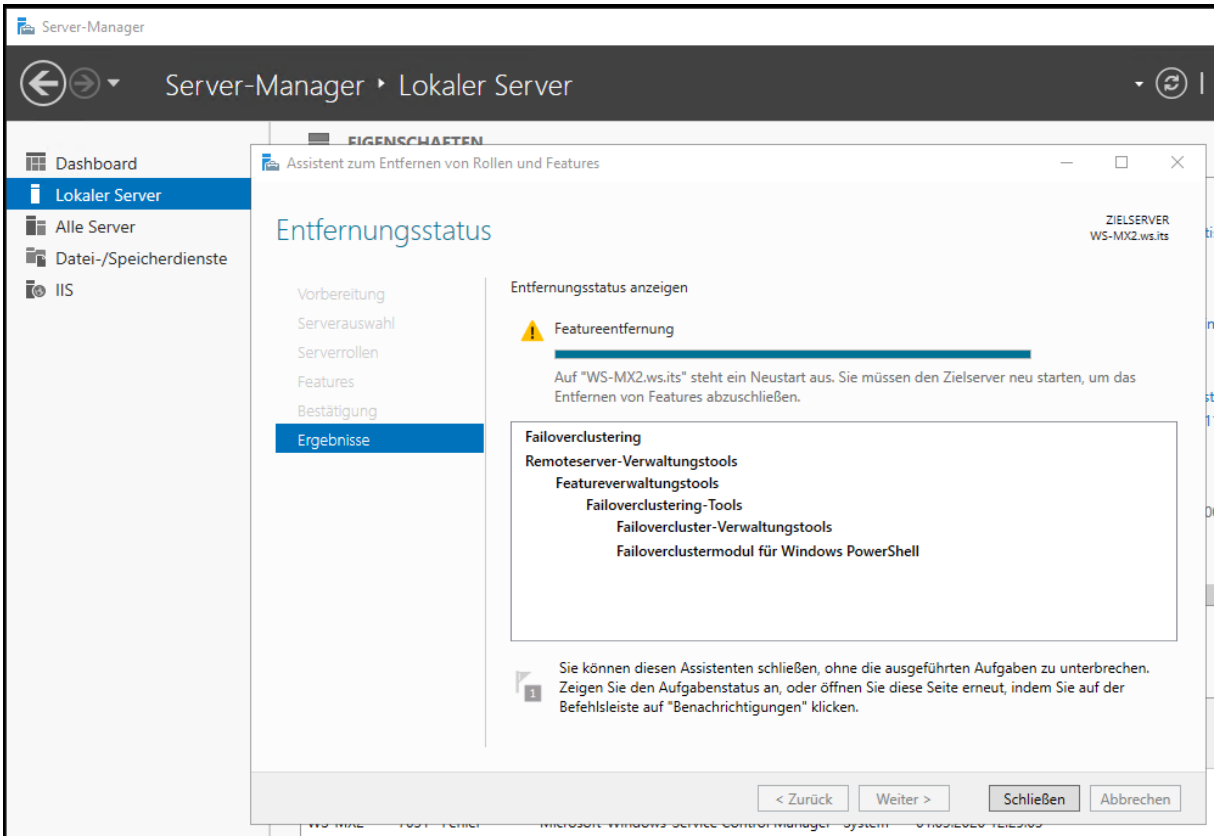
Der Cluster-Dienst lässt sich leider immer noch nicht starten:



Daher versuche ich es jetzt mit einer Deinstallation des Features „Failover-Cluster“:



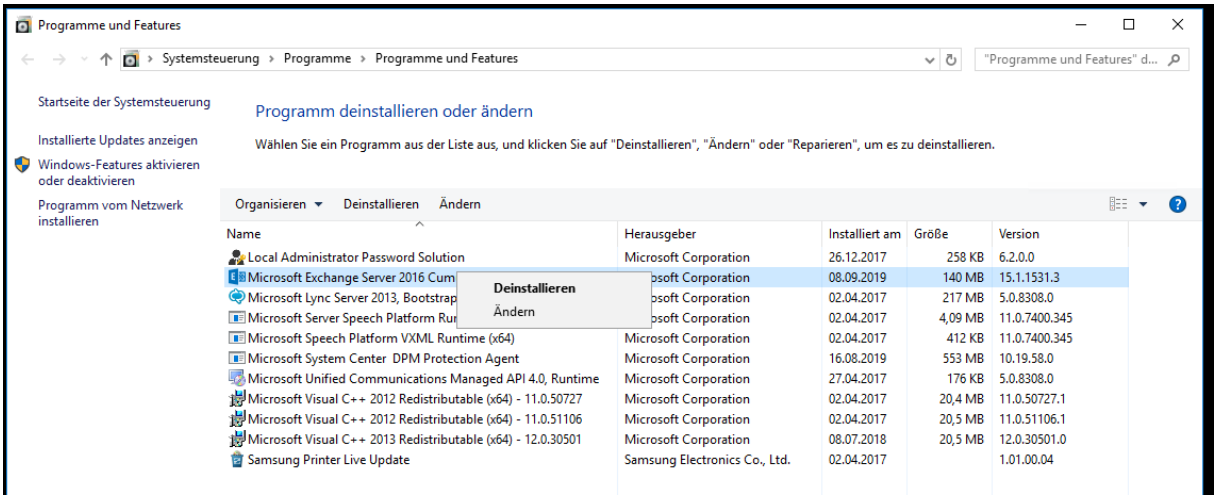
Interessanterweise hat die Entfernung funktioniert:



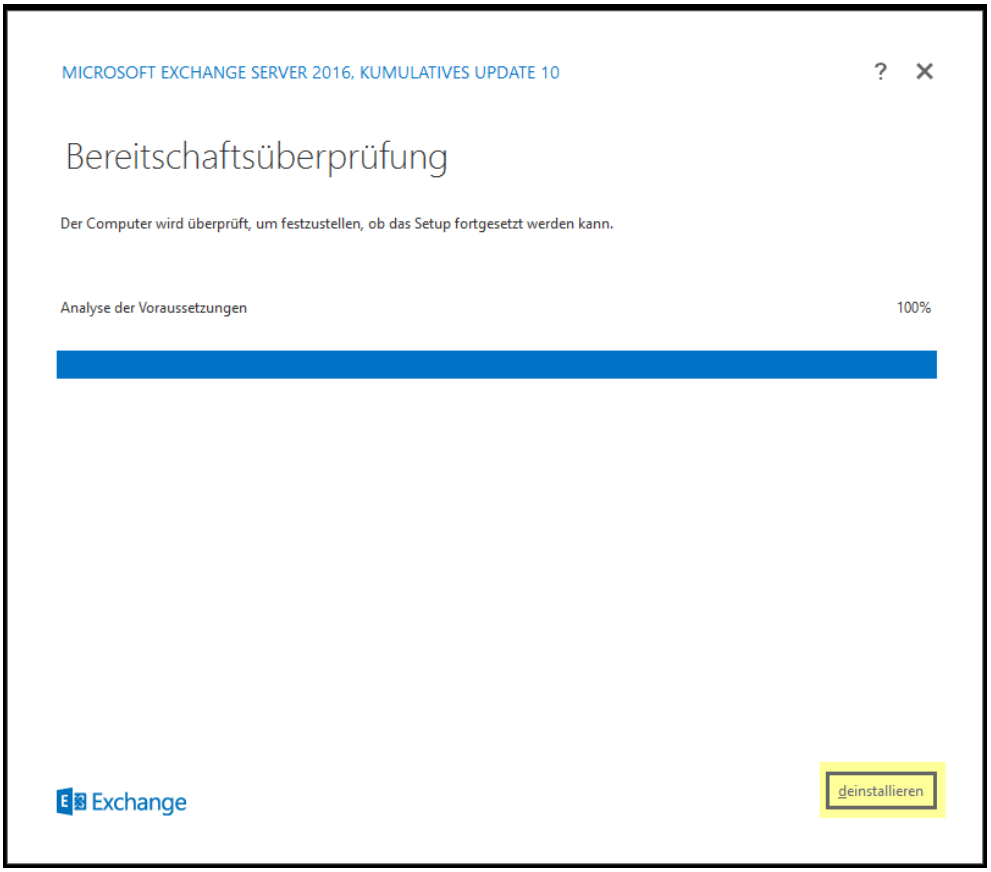
Es ist für den Abschluss ein Neustart erforderlich.

Deinstallation des Exchange Servers – Problem: Memory Leak

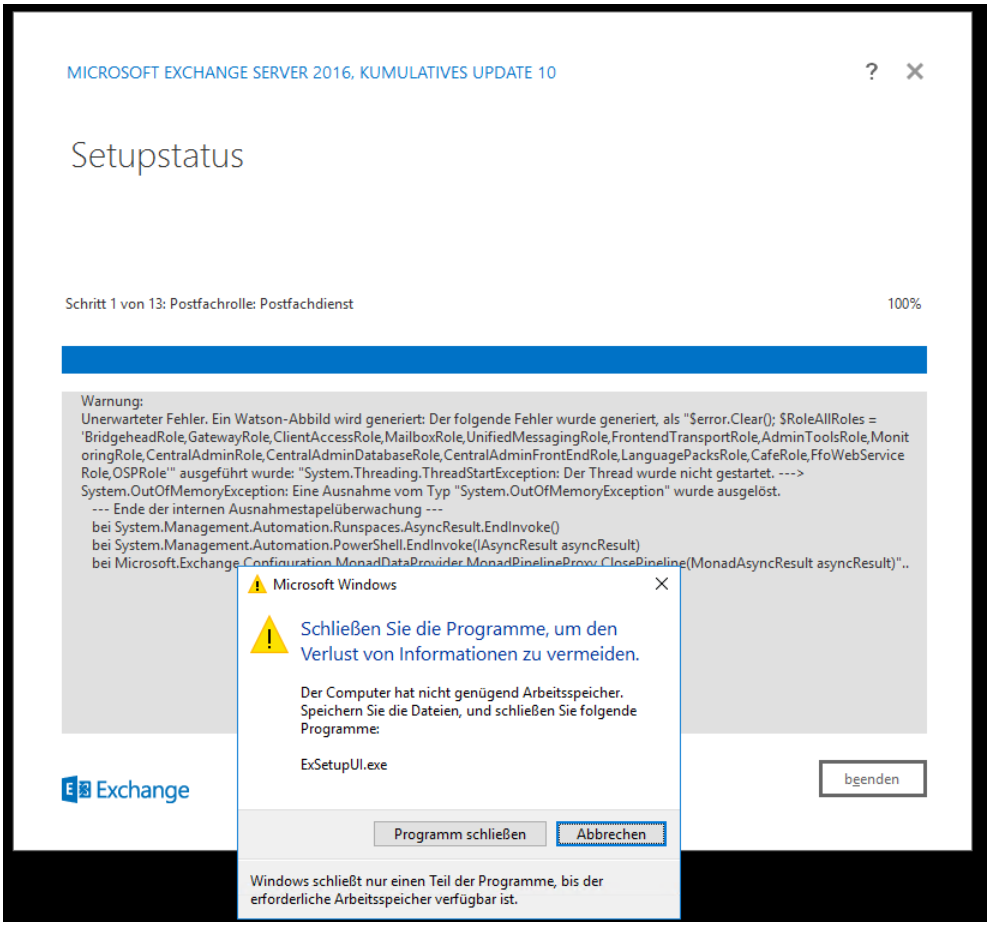
Ich versuche erneut mit der Deinstallation:



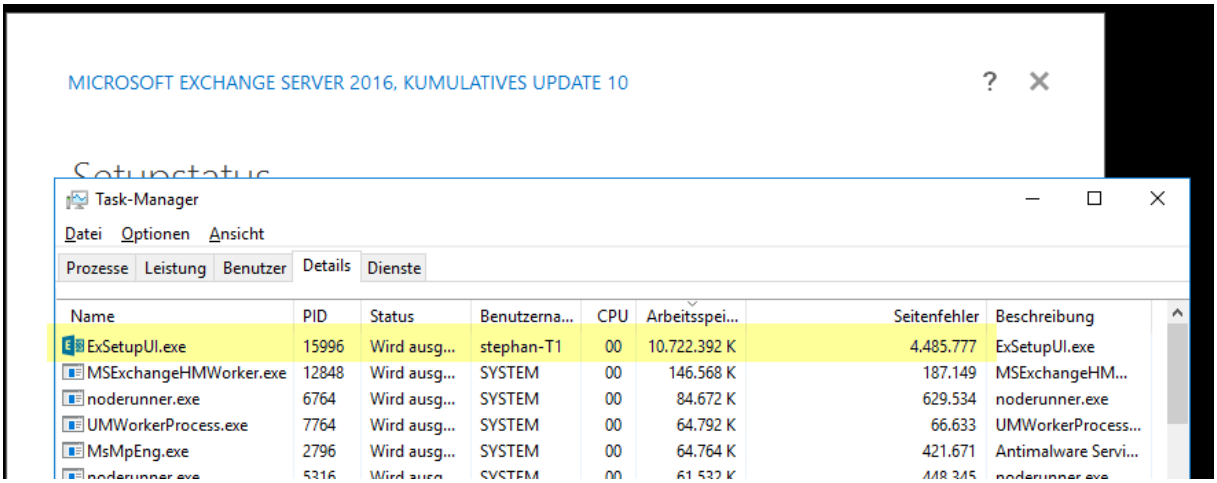
Wieder startet der Assistent und sucht nach verbliebenen Abhängigkeiten. Das Entfernen des Clusters hat funktioniert! Die Deinstallation kann gestartet werden:



Leider ist die Freude von sehr kurzer Dauer. Das System hat keinen ausreichenden Arbeitsspeicher:

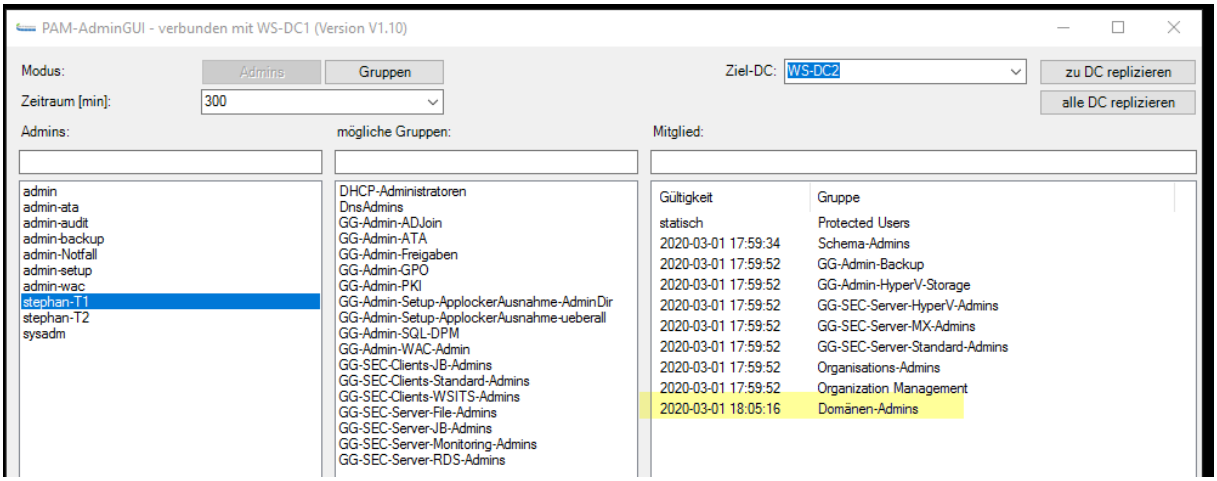


Eigentlich ist der Server mit 14GB RAM ausgestattet und hat keine Last mehr. Das sollte doch genügen. Wo also wird der Arbeitsspeicher verbraucht? Ha: das Setup selber belegt 10GB!!!

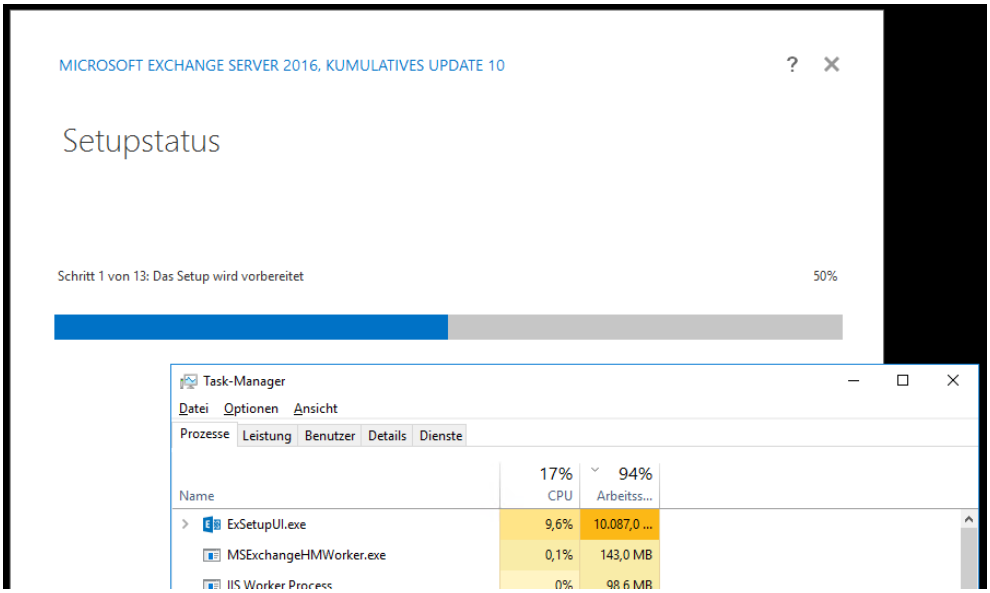


Deinstallation des Exchange Servers

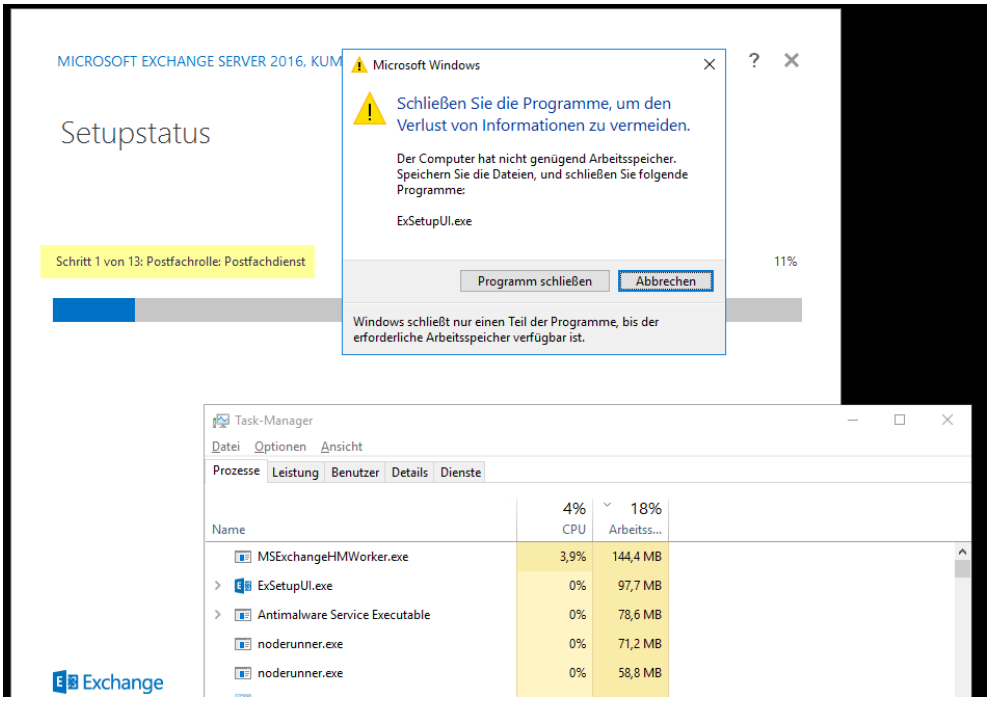
Ich nutze aber die Gelegenheit der gescheiterten Deinstallation und erweitere die Berechtigungen meines administrativen Accounts. Diese hatte ich zuvor vergessen:



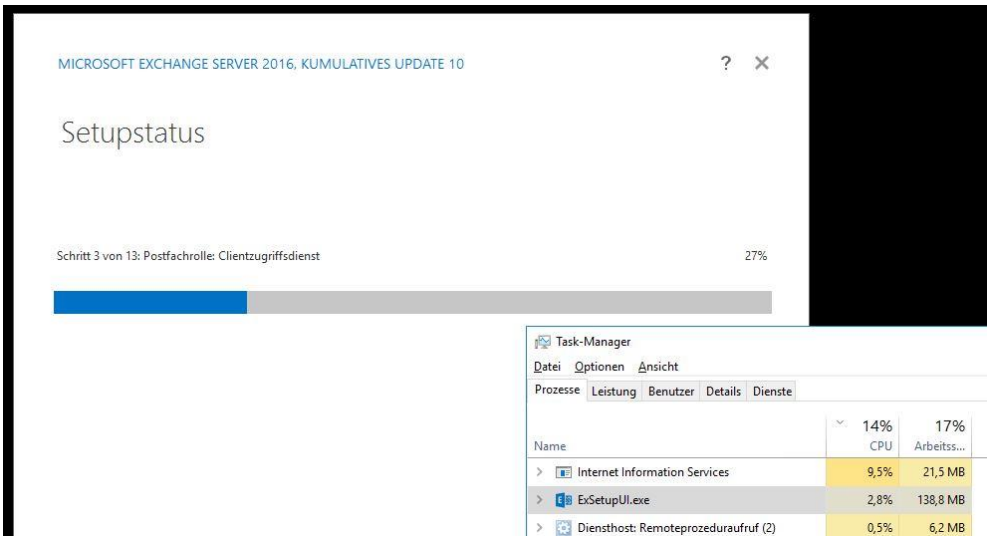
Nach einer Neuanmeldung probiere ich es erneut. Dieses Mal läuft das Setup weiter. Der Prozess selber bindet wieder innerhalb von Sekunden fast den gesamten Arbeitsspeicher:



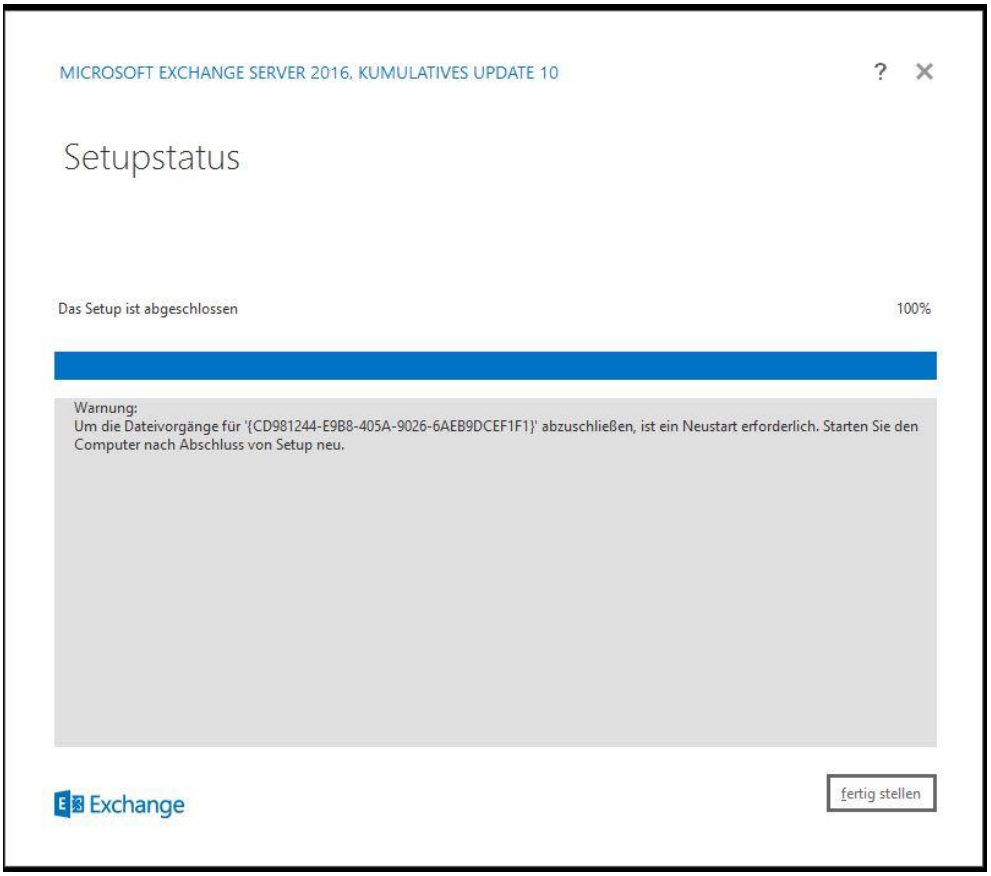
Die Meldung vom Betriebssystem lässt nicht lange auf sich warten. Aber das Setup läuft weiter und gibt auf einen Schlag den Arbeitsspeicher wieder frei. Was soll man davon halten…



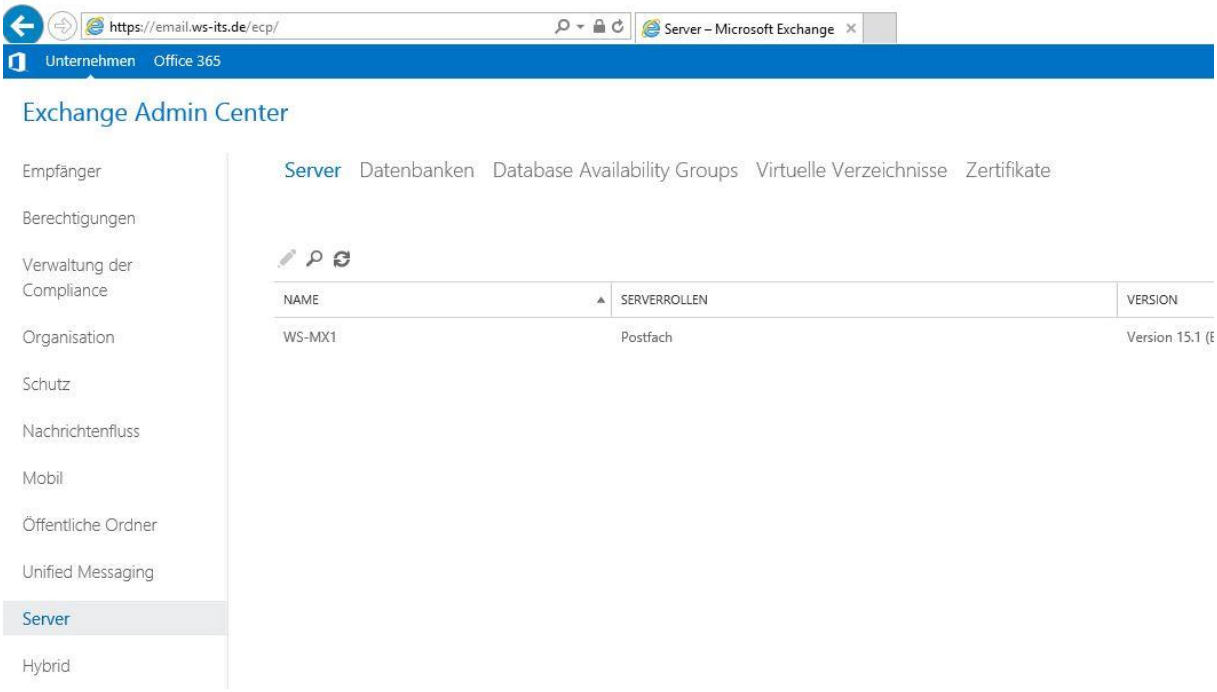
Jetzt kommt die Deinstallation voran:



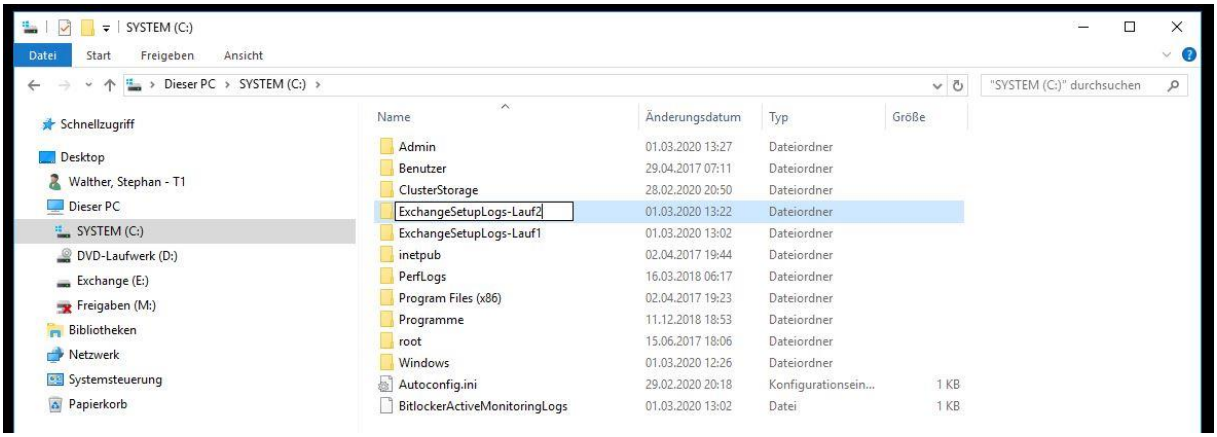
Nach einigen Minuten ist der Vorgang abgeschlossen. Ein Neustart steht aus:



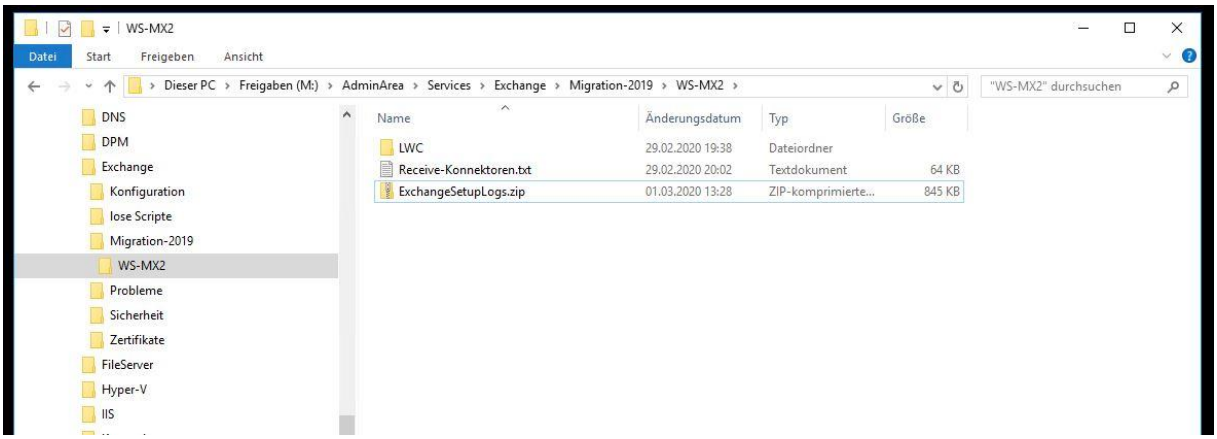
Während der Server WS-MX2 neustartet, kontrolliere ich im Exchange Admin Center die Lage. Der Server wird nicht länger angezeigt:



Nachdem der Server seinen Neustart abgeschlossen hat, archiviere ich die Logfiles des Setups. Falls doch etwas schief gelaufen ist, kann ich hier im Nachgang den Fehler suchen:



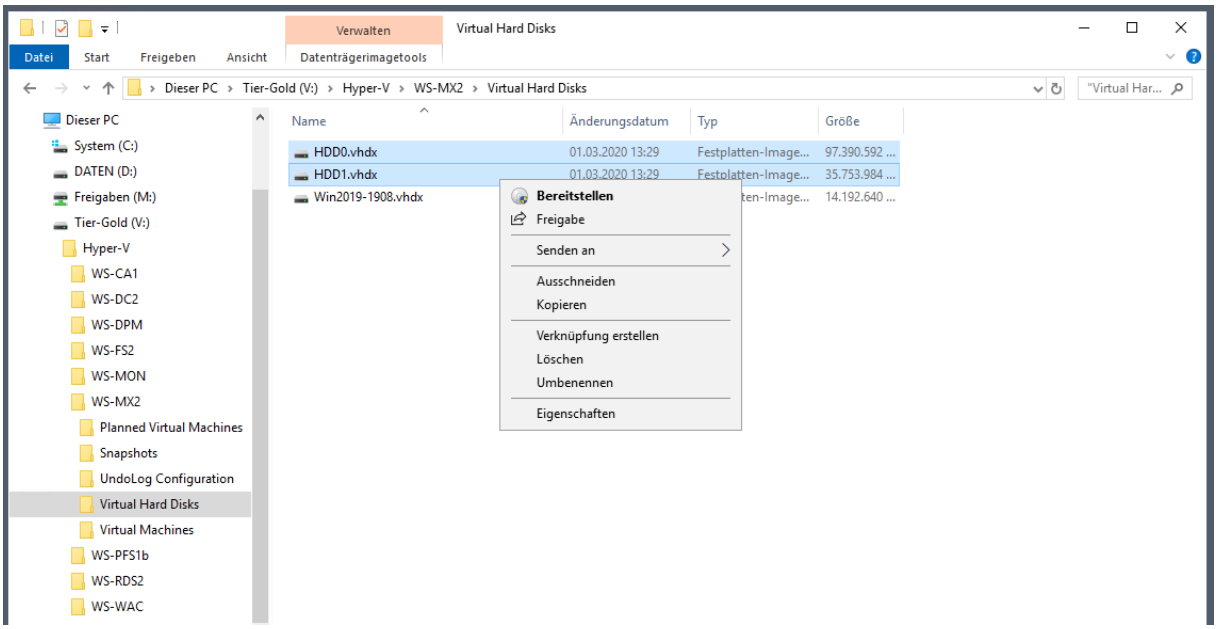
Das Logfile-Archiv verschiebe ich in meine administrative Freigabe:



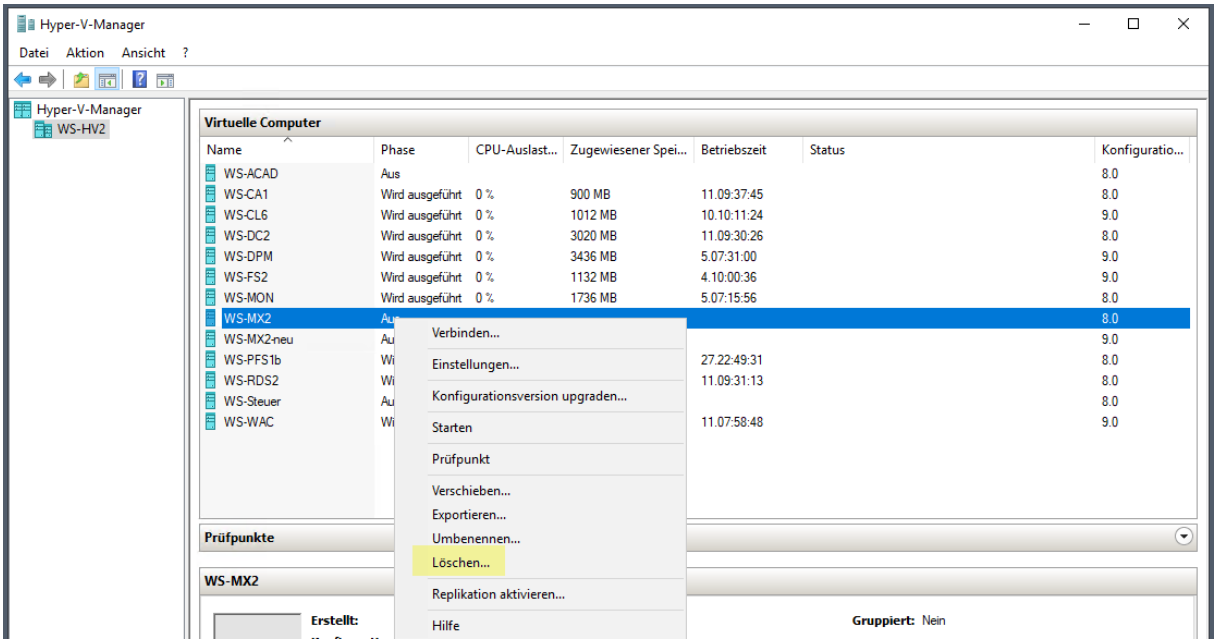
Jetzt ist auf dem Server WS-MX2 nichts mehr vorhanden.

Entfernung des alten Servers und Austausch der VM

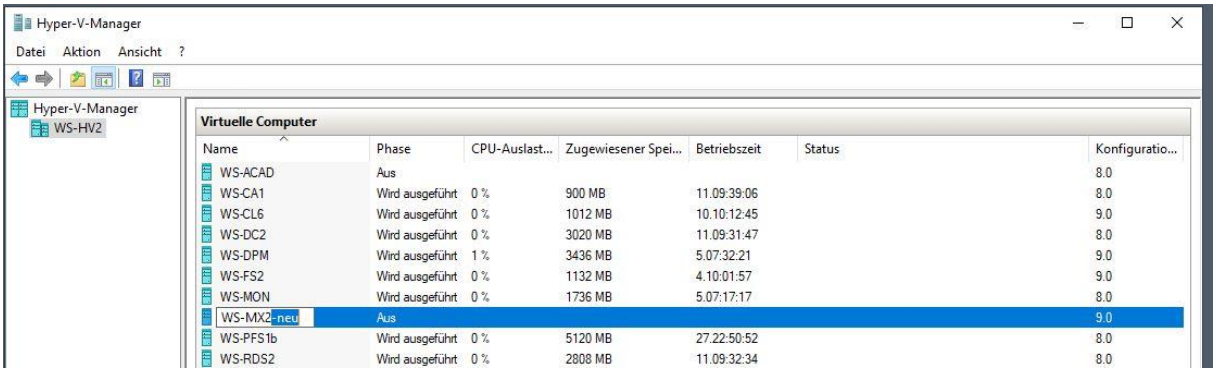
Ich fahre den Server herunter und entferne ihn im Hyper-V. Dazu lösche ich die virtuellen Festplatten-Dateien…



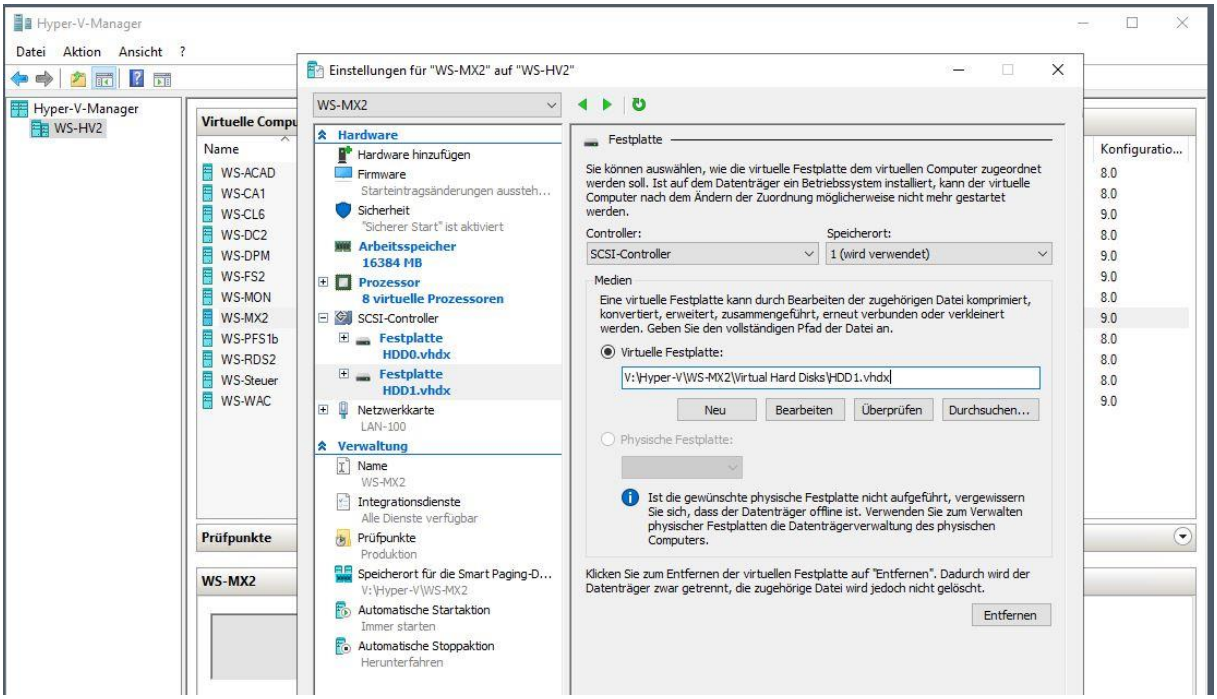
… ebenso wie die virtuelle Maschine selber:



Der Name WS-MX2 ist nun frei. Daher benenne ich die vorbereitete virtuelle Maschine um:



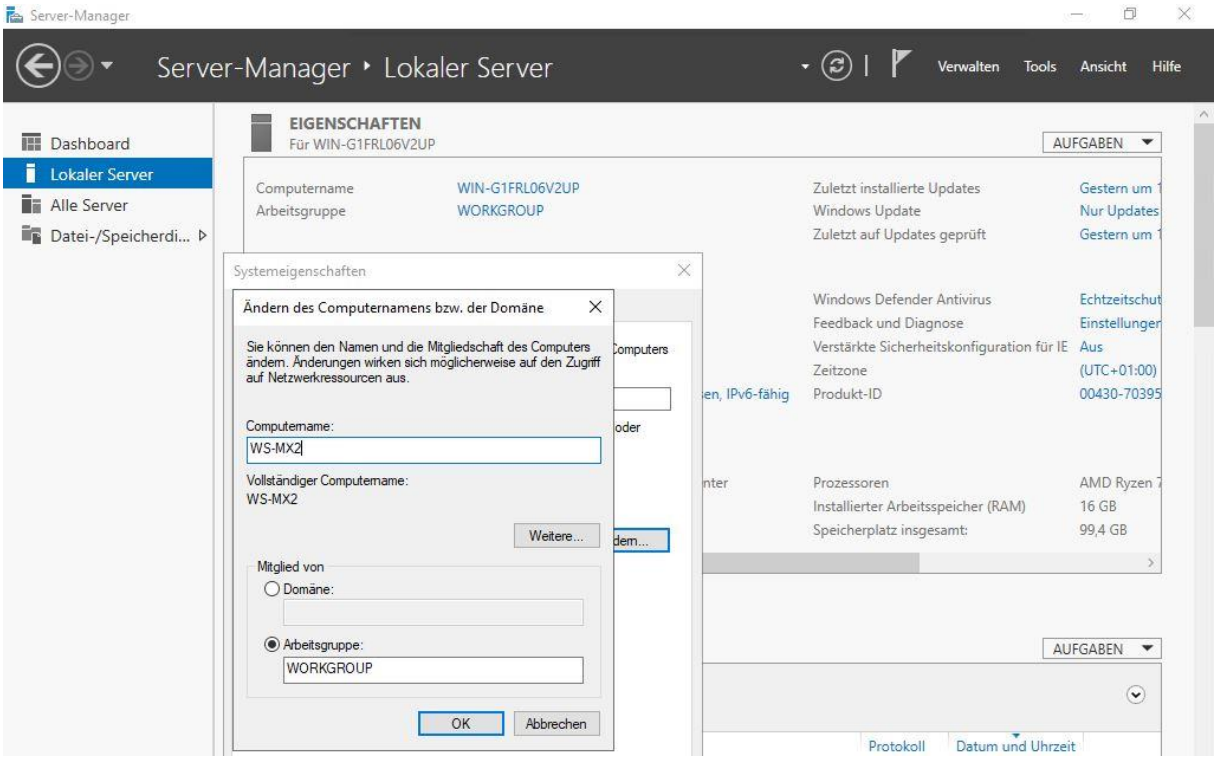
Der neue Server erhält weitere Ressourcen. 16GB, 8 vCPU und eine neue virtuelle Festplatte für die Datenbanken kommen dazu:



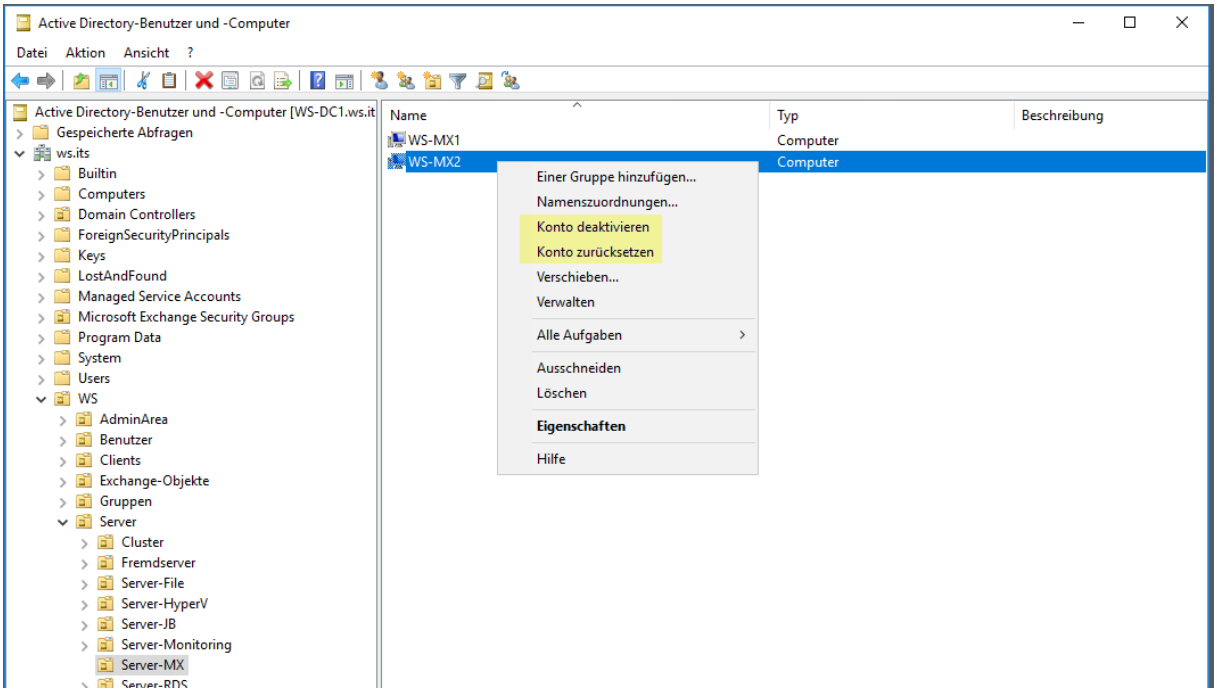
Bereitstellung des neuen Mailservers (MX2019)

Grundkonfiguration des Betriebssystems

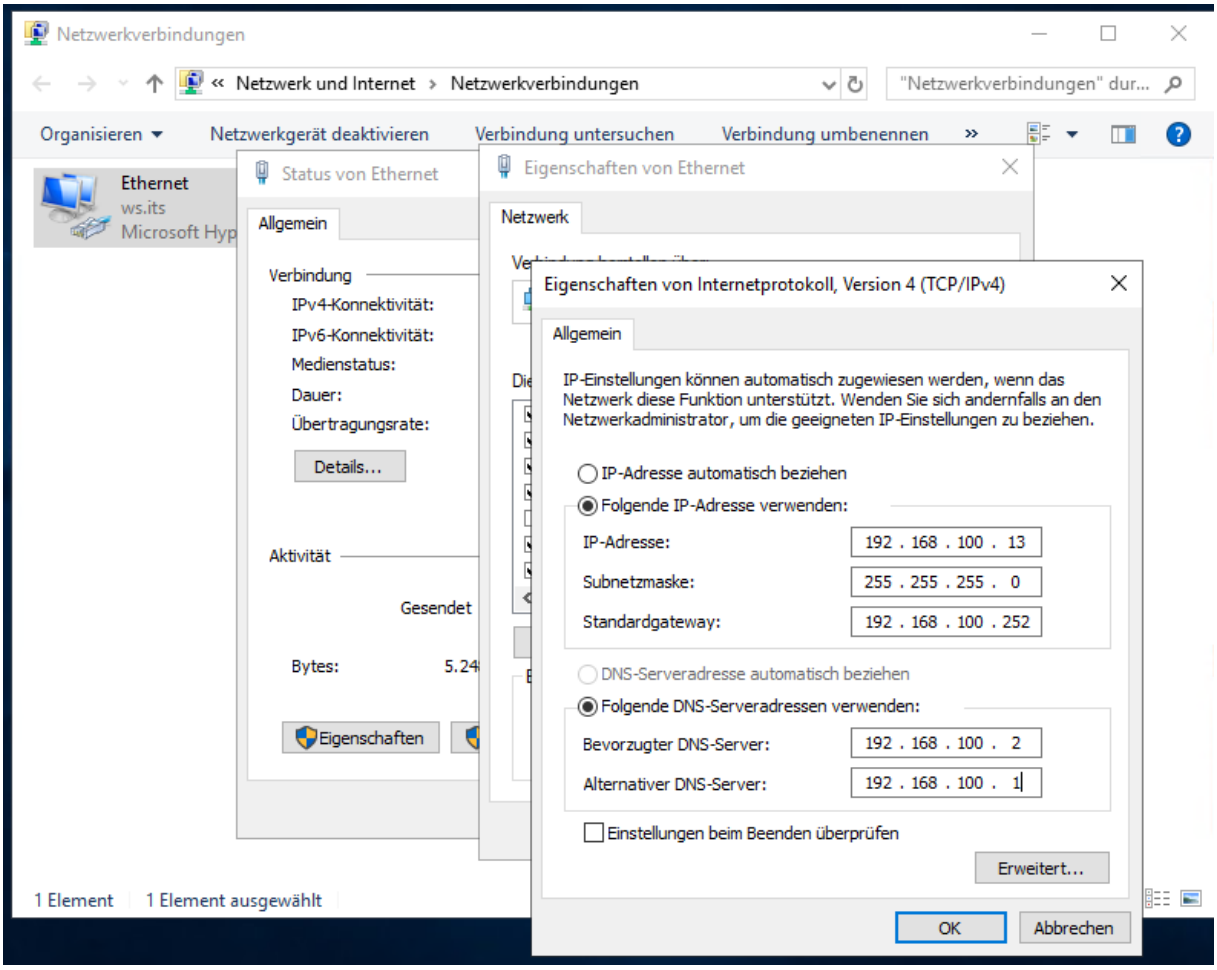
Ich starte die neue VM. Zuerst benenne ich den Server um. Den Namen WS-MX2 verwende ich weiter:



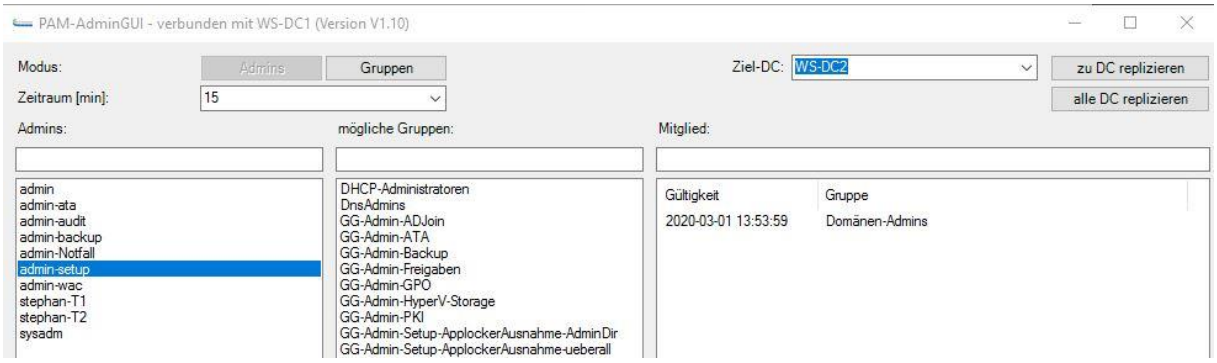
Wie üblich ist ein Neustart erforderlich. Während dieser Zeit bereite ich das Active Directory Computerkonto für die Übernahme vor:



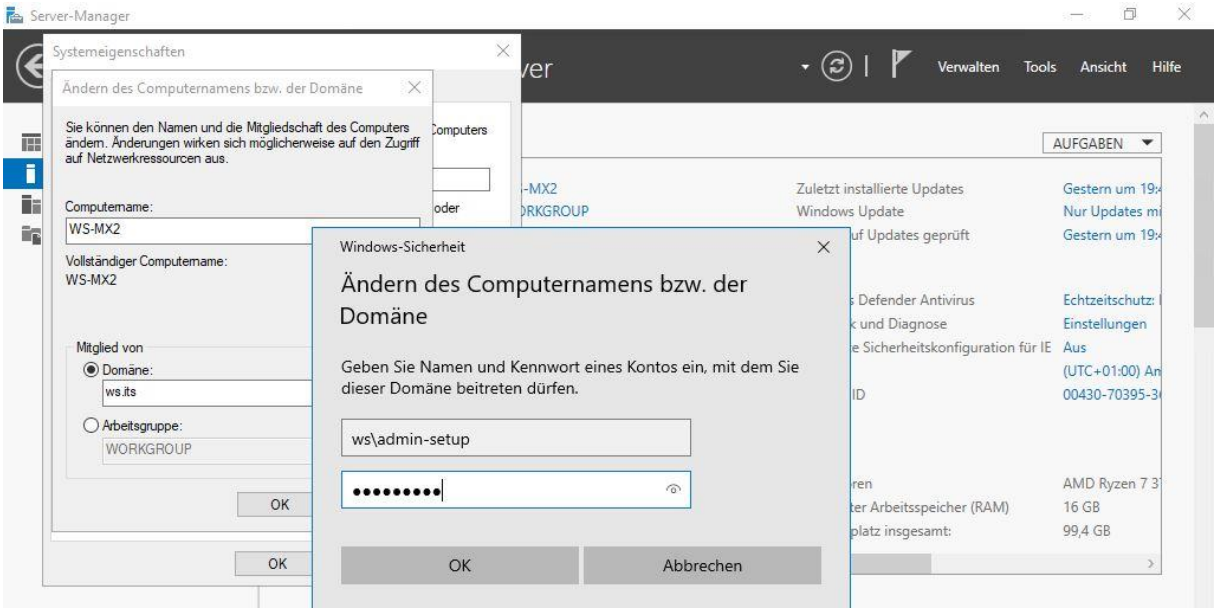
Wieder angemeldet passe ich die IPv4-Konfiguration an. Auch die IP-Adresse 192.168.100.13/24 verwende ich wieder. So spare ich mir die Anpassungen in der Firewall und im LoadBalancer:



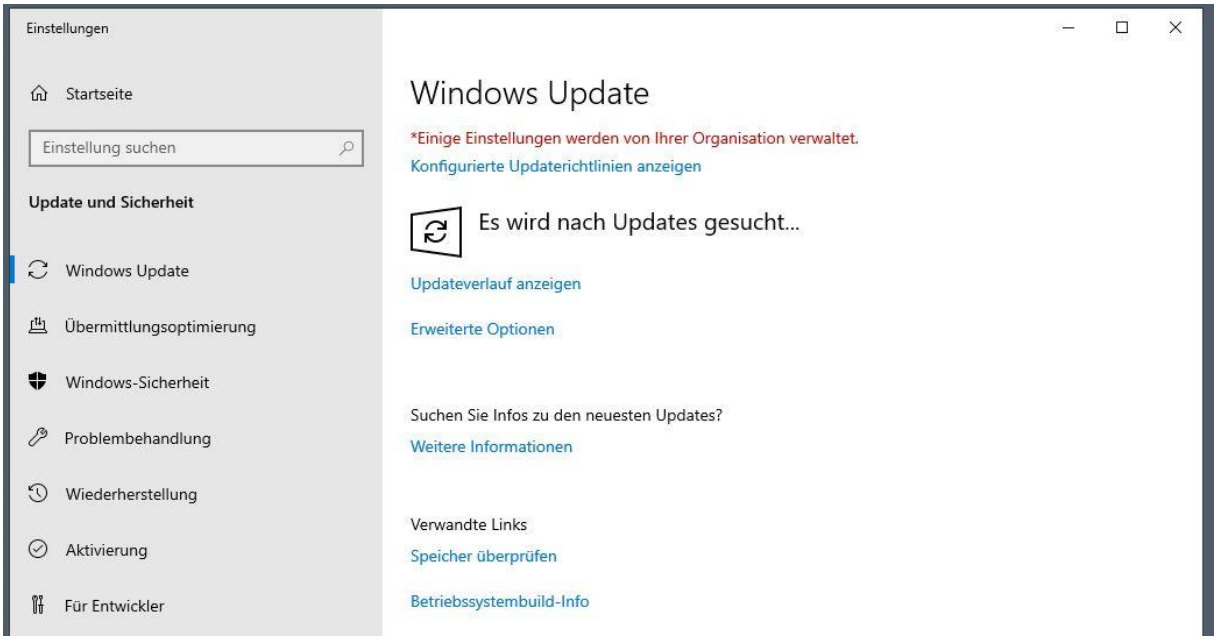
Für den Domain Join bereite ich einen Account vor:



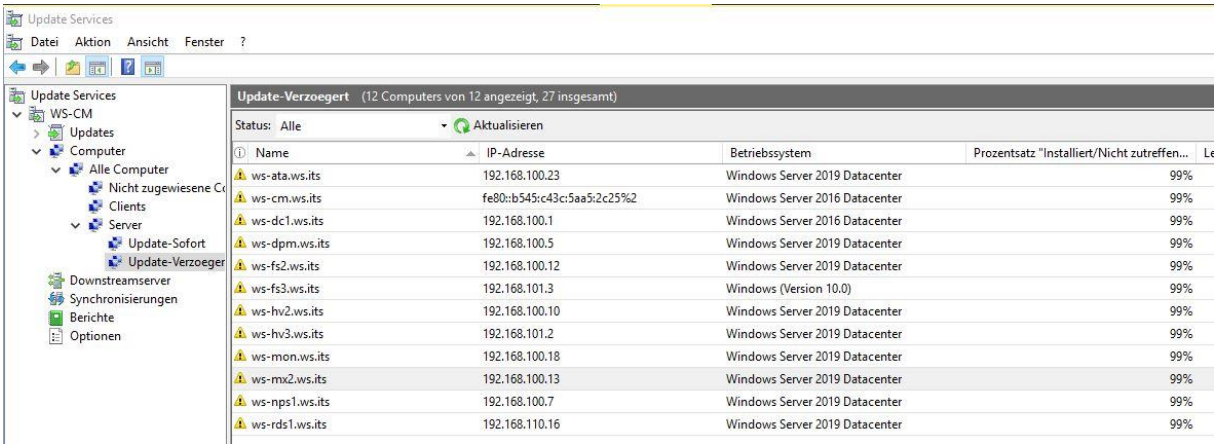
15 Minuten sind dafür mehr als ausreichend. Ich nehme den neuen Server in die Domain auf. Dabei übernimmt er das Computerkonto und somit die Identität des alten Mailservers:



Nach einem weiteren Neustart werden die Gruppenrichtlinien angewendet. Ich starte noch einmal ein Windows Update. Dieses Mal kommen die Daten von meinem WSUS:

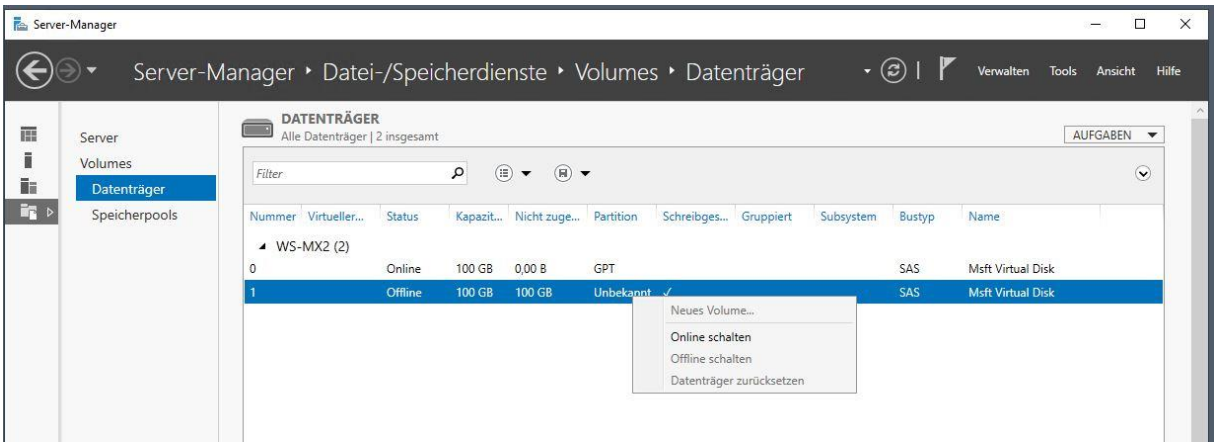


Durch die manuelle Aktivierung der Aktualisierung meldet sich der Server im WSUS:

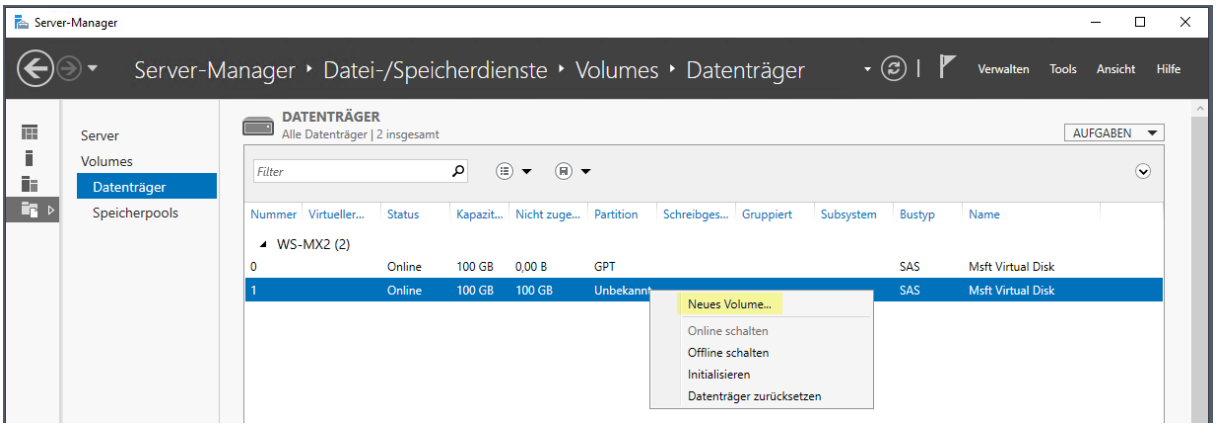


Interessanterweise werden Updates auf dem WSUS gefunden. Eigentlich war ja schon alles installiert. Aber ok – dann installiere ich diese eben noch einmal:

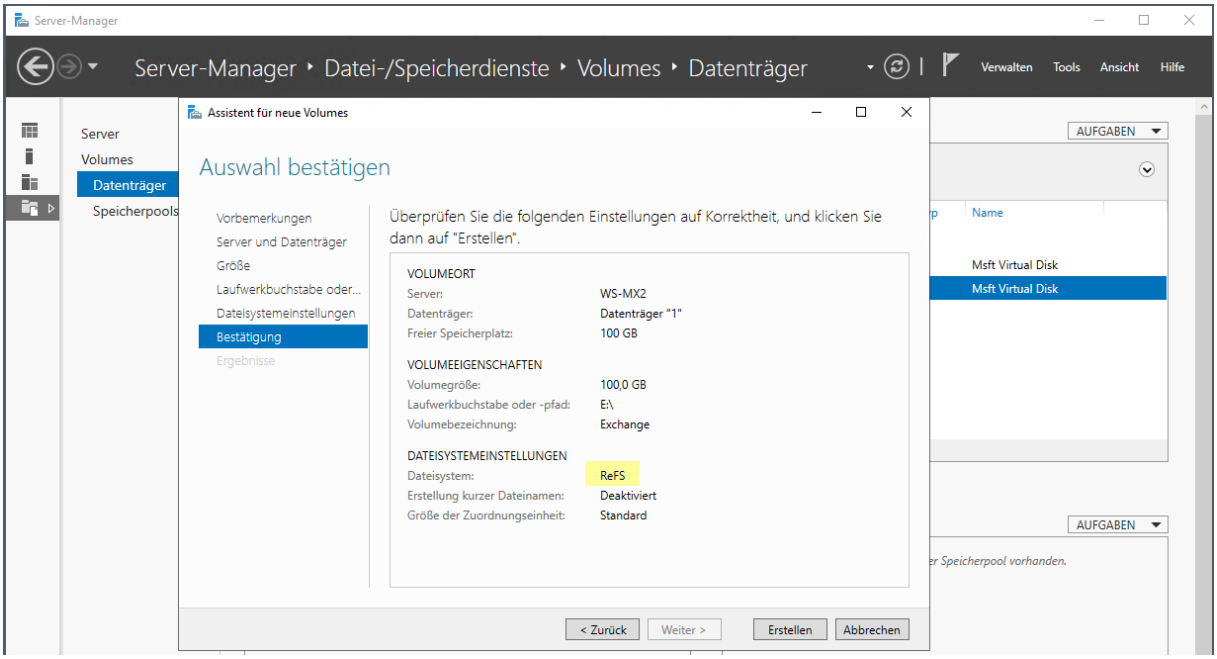
Während der Aktualisierung kümmere ich mich um die neue Festplatte. Darauf erstelle ich eine neue Partition. Zuerst muss sie aber aktiviert werden:



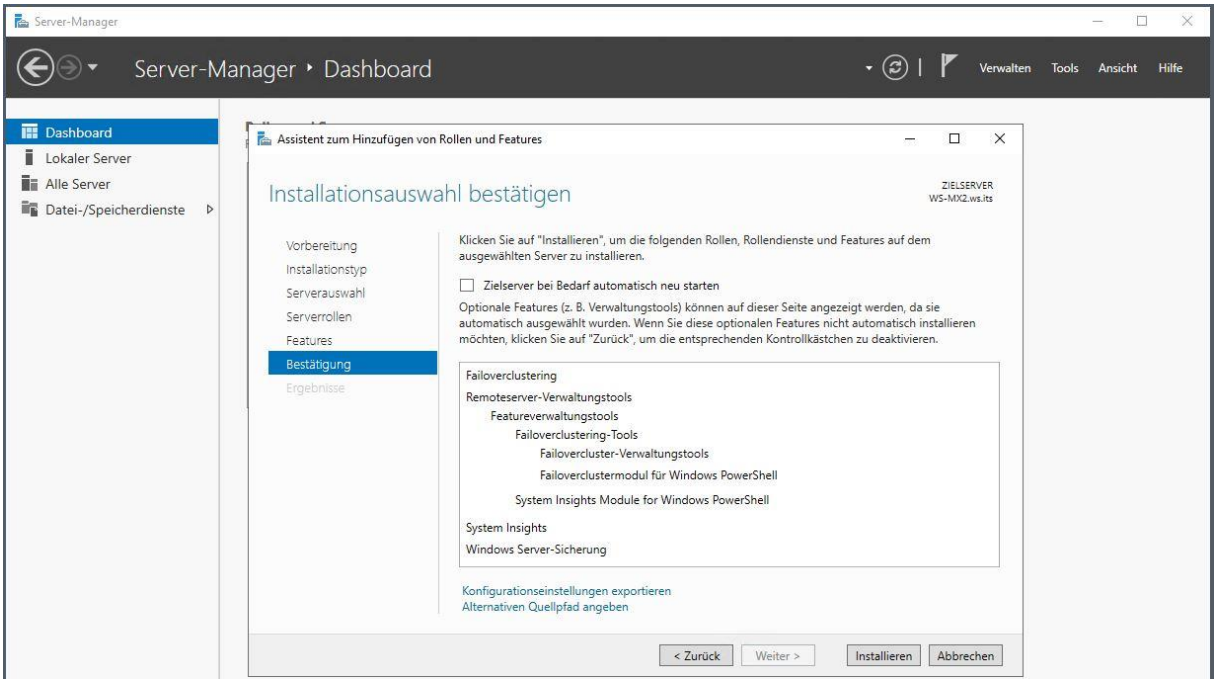
Das neue Volume kann einfach mit dem Server-Manager erstellt werden:



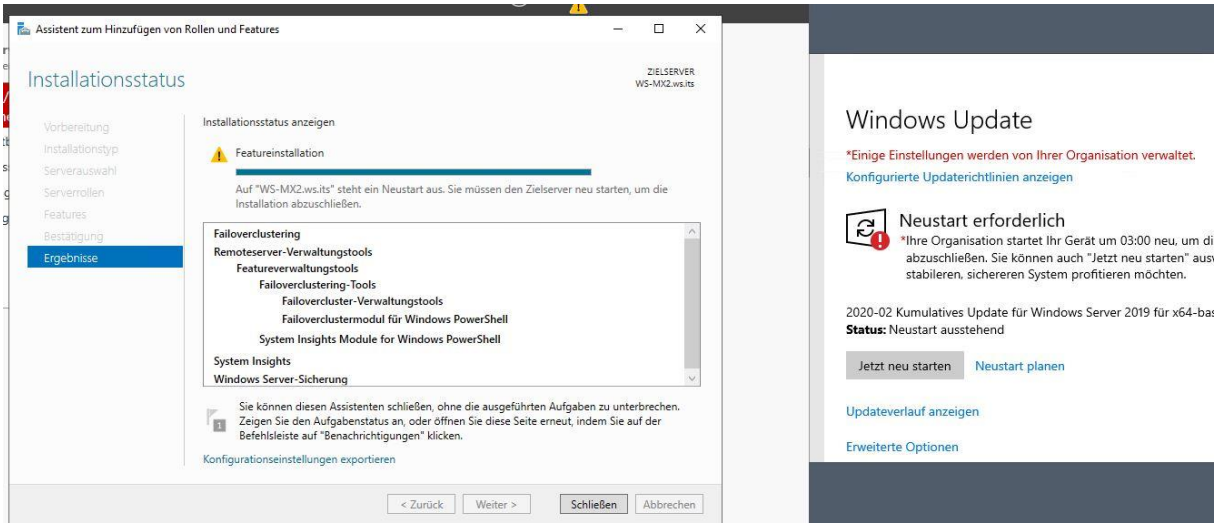
Als Dateisystem verwende ich das von Microsoft für Exchange Server 2016+ empfohlende ReFS. Meine Datensicherung ist damit kompatibel:



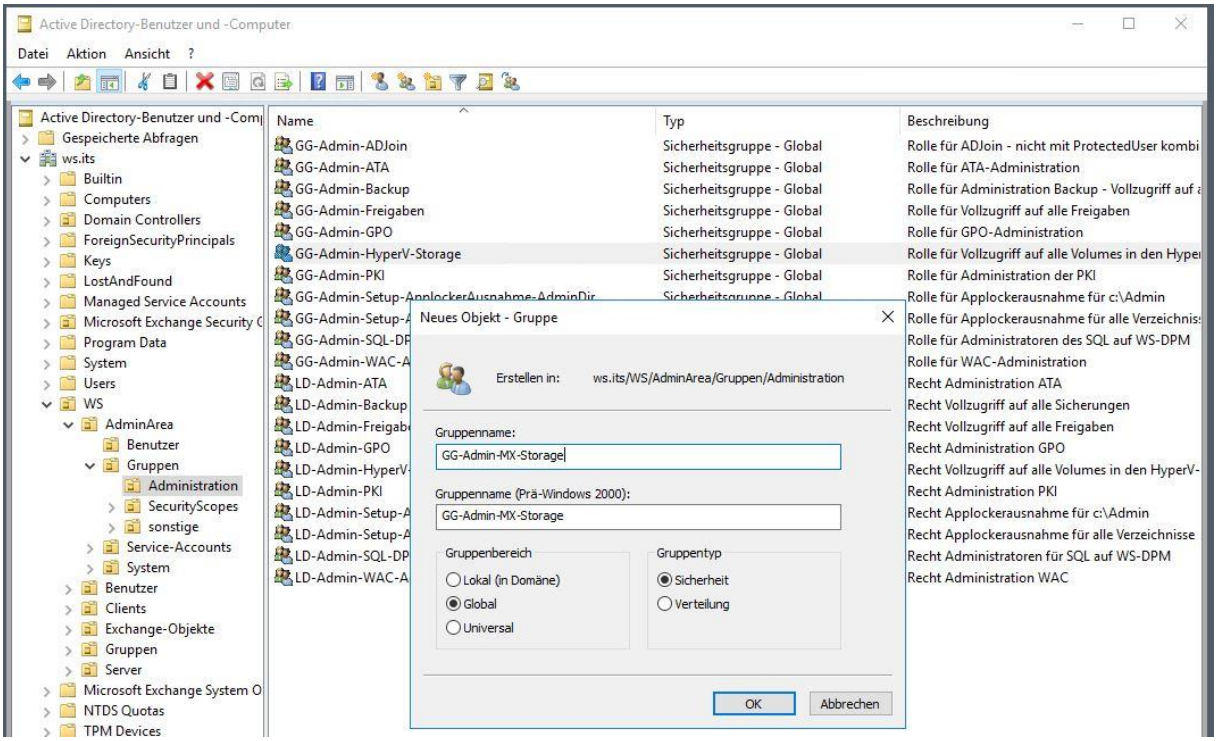
Jetzt installiere ich noch ein paar Rollen und Features, die nicht zwingend zum Exchange Server gehören. Mit System Insights kann ich den Trend der Serverbelastung später mit dem Windows Admin Center überwachen. Und die Windows Server Sicherung soll später für die Datensicherung des SystemStates verwendet werden:



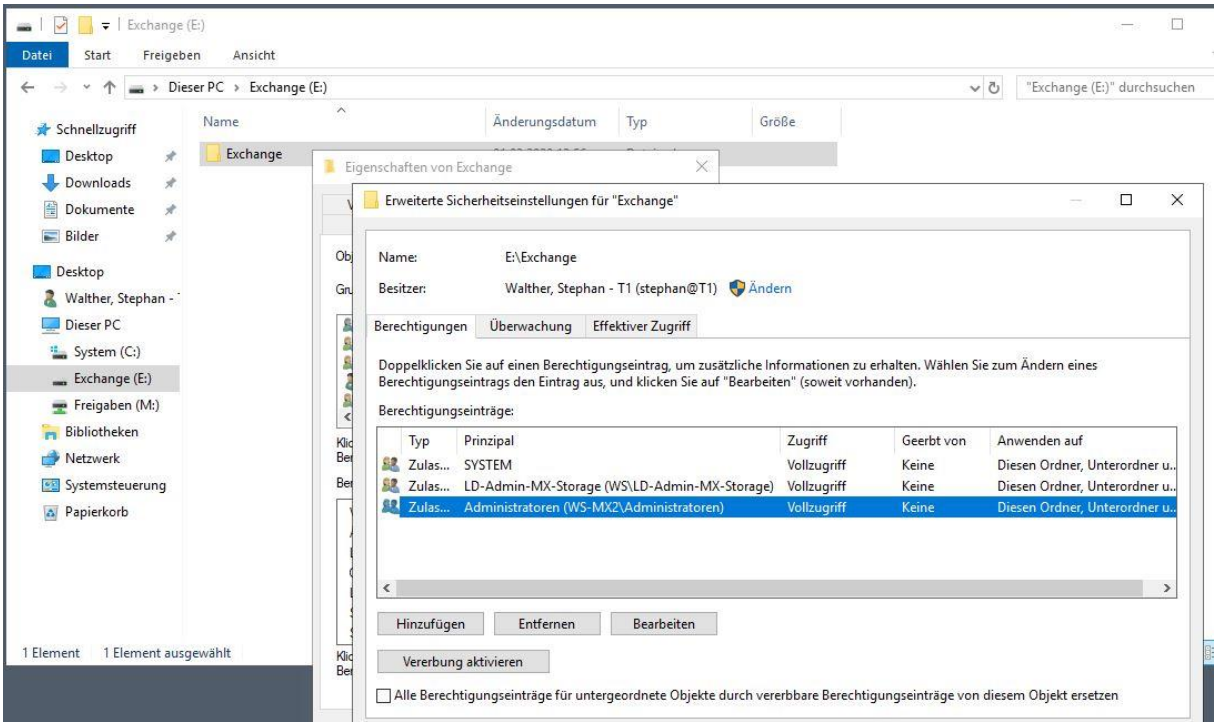
Die Rollen- und Feature-Installation und die Windows Updates erfordern einen Neustart:



Für den Zugriff auf das neue Volume erstelle ich im Active Directory zwei neue Gruppen GG-Admin-MX-Storage und LD-Admin-MX-Storage. Die GG ist in der LD als Mitglied verschachtelt:



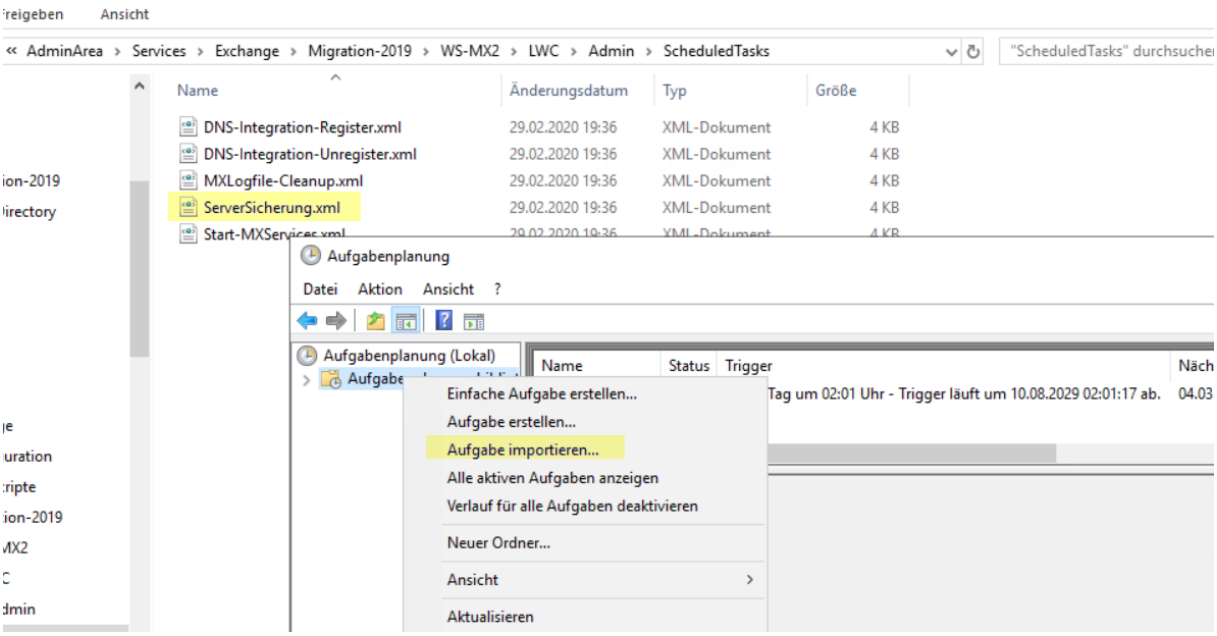
Auf dem neuen Volume ändere ich die Stammberechtigung ab. Damit ist ein einfacher Zugriff auf das Volume nicht mehr möglich:



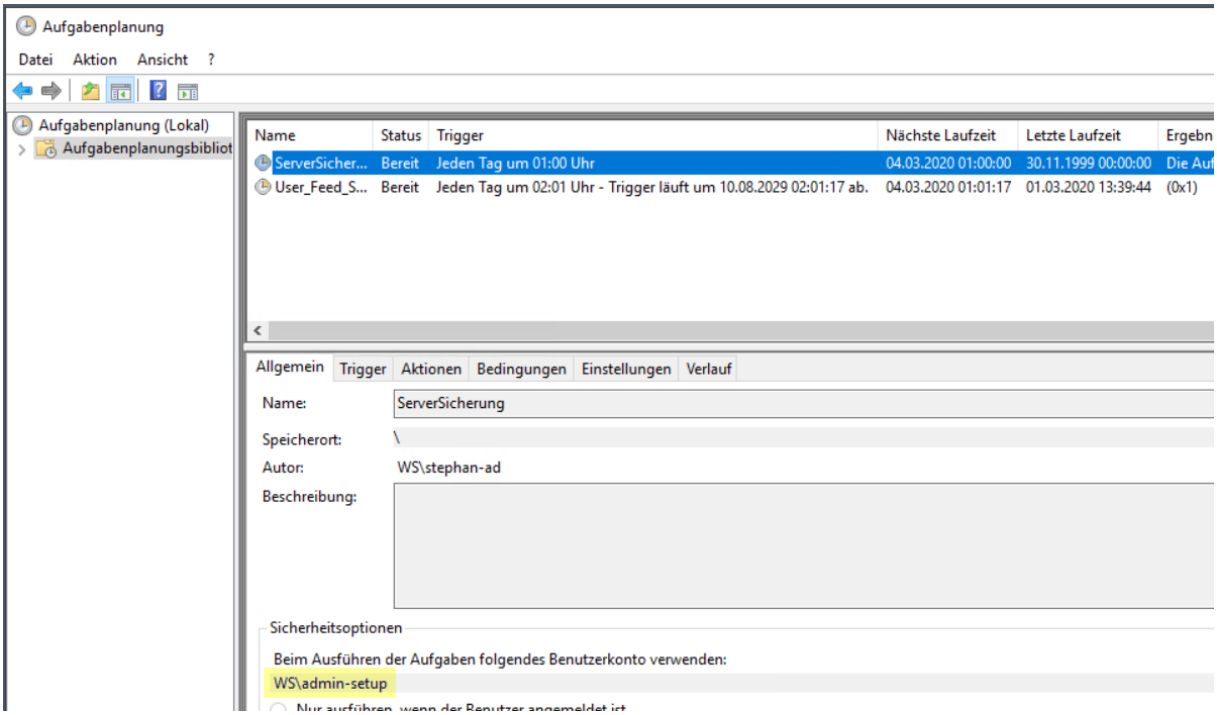
Diese Konfiguration ist für mein Rollenzugriffs-Konzept gedacht. So kann ich später den Zugriff auf die Datenbank-Partition explizit delegieren.

Einrichtung der Datensicherung (BMR mit Windows Server Sicherung)

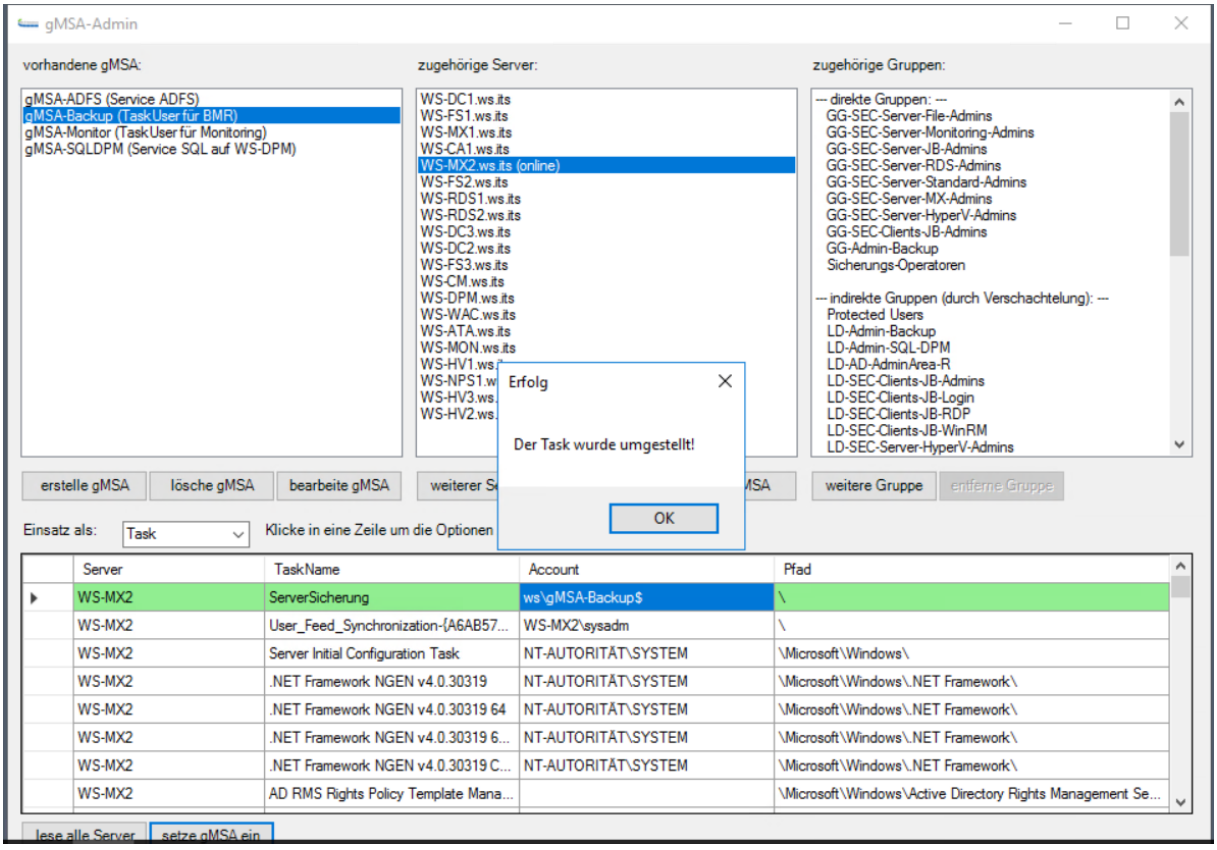
Bevor der Server in die produktive Phase geht konfiguriere ich die Serversicherung. Diese ist wie bei meinen anderen Servern auch auf 2 Strategien aufgebaut: Das Betriebssystem ziehe ich als SystemImage mit der Windows Server Sicherung ab. Nutzdaten – wie Datenbanken – sichere ich mit dem Data Protection Manager. Hier konfiguriere ich die Serversicherung. Diese wird durch eine geplante Aufgabe gestartet. Als XML-Datei kann ich sie recht einfach importieren:



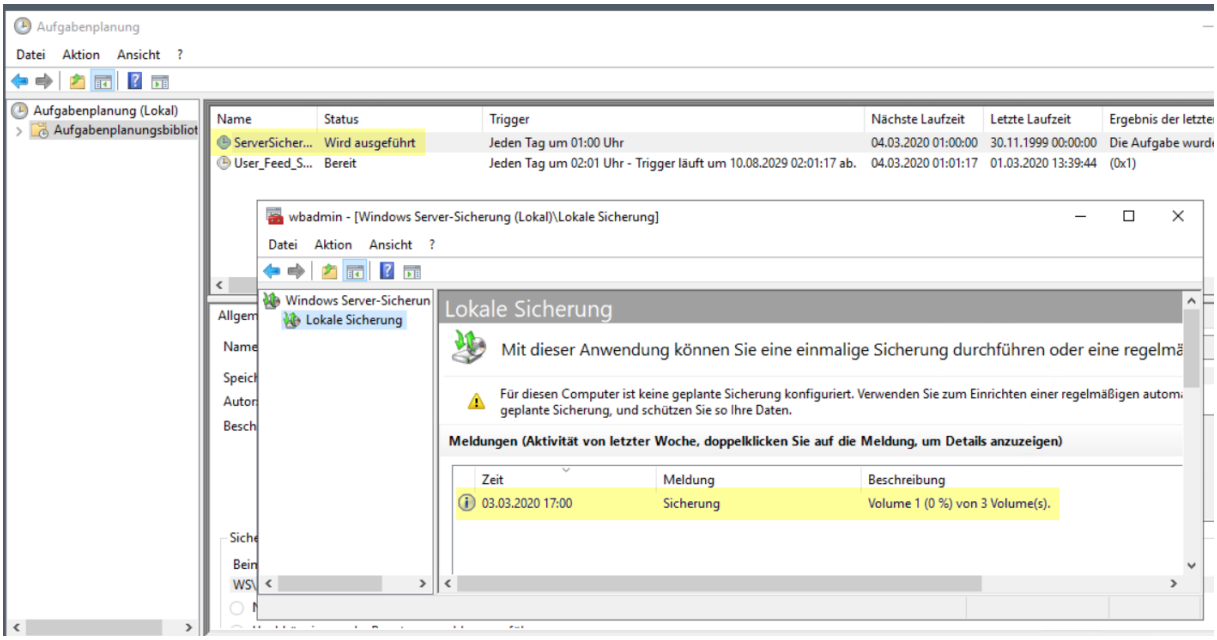
Der Sicherungstask wird mit einem Group Managed Service Account (gMSA) ausgeführt. Beim Speichern der Aufgabe habe ich einen Dummy-Account eingetragen, da ich das Passwort des gMSA nicht kenne – die Server holen sich diese Information geschützt vom Domain Controller. Aber ohne Passwort kann ich den Task nicht speichern:



Mit meiner PowerShell-GUI „gMSA-Admin“ konfiguriere ich nun den Sicherungstask remote vom Domain Controller aus neu:



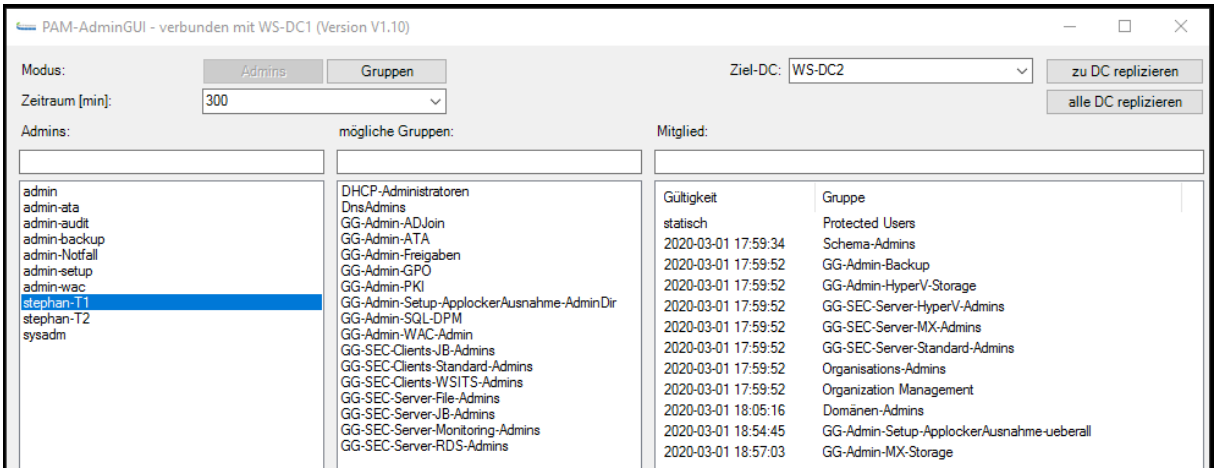
Bevor ich mit der Installation beginne, starte ich die Datensicherung:



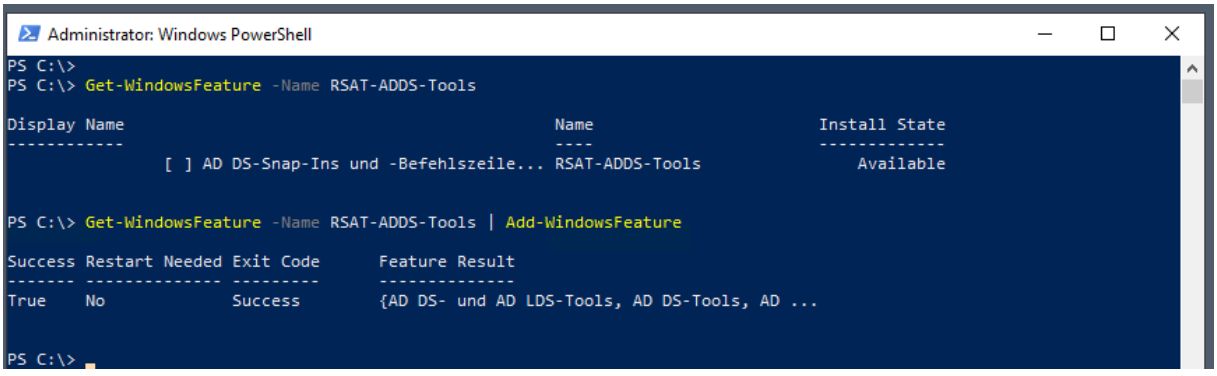
Sie dauert nur wenige Minuten. Mit dieser Sicherung kann ich ein Rollback durchführen, wenn bei der Installation vom Exchange Server etwas schief läuft.

Vorbereitung des AD für Exchange Servers 2019 CU4

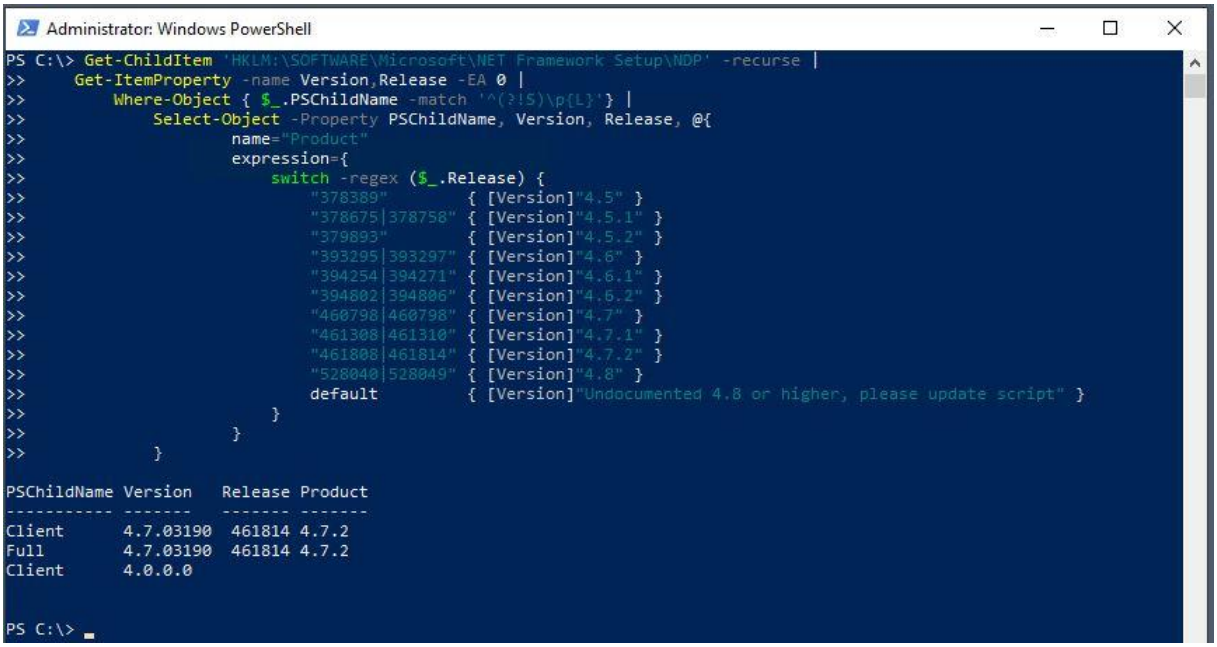
Auf WS-MX2 installiere ich den ersten Exchange Server 2019 mit dem kumulativen Update 4. Dabei ist im Normalfall immer eine Vorbereitung des Active Directory erforderlich. Also statte ich meinen administrativen Account temporär mit den richtigen Gruppenmitgliedschaften aus. Jetzt ist er Mitglied in den Gruppen „Schema-Admins“ und „Organisations-Admins“ (engl. „Enterprise-Admins“):



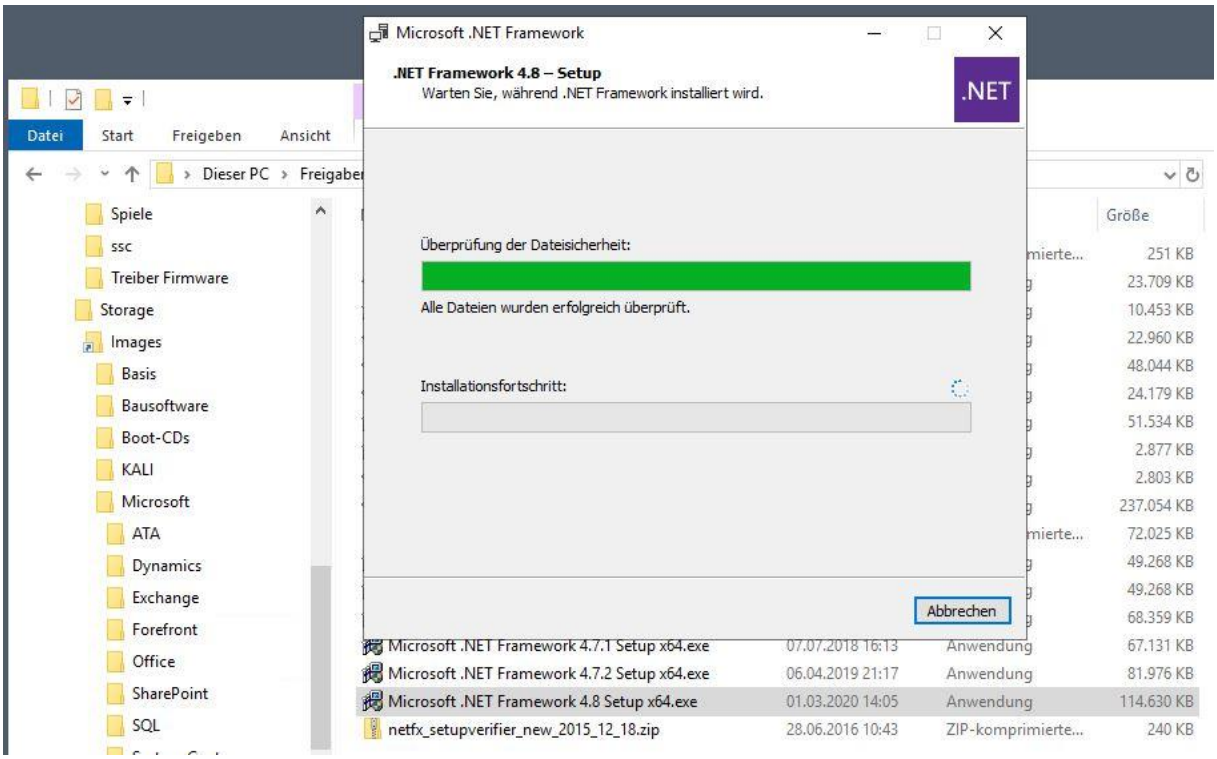
Die Aktualisierung des Active Directory starte ich direkt vom zukünftigen Exchange Server WS-MX2 aus. Dafür werden die RSAT-Features für das Active Directory erforderlich. Diese installiere ich mit der PowerShell:



Eine weitere Voraussetzung ist das aktuelle .net-Framework 4.8. Dieses ist selbst auf einem Windows Server 2019 nicht standardmäßig installiert:

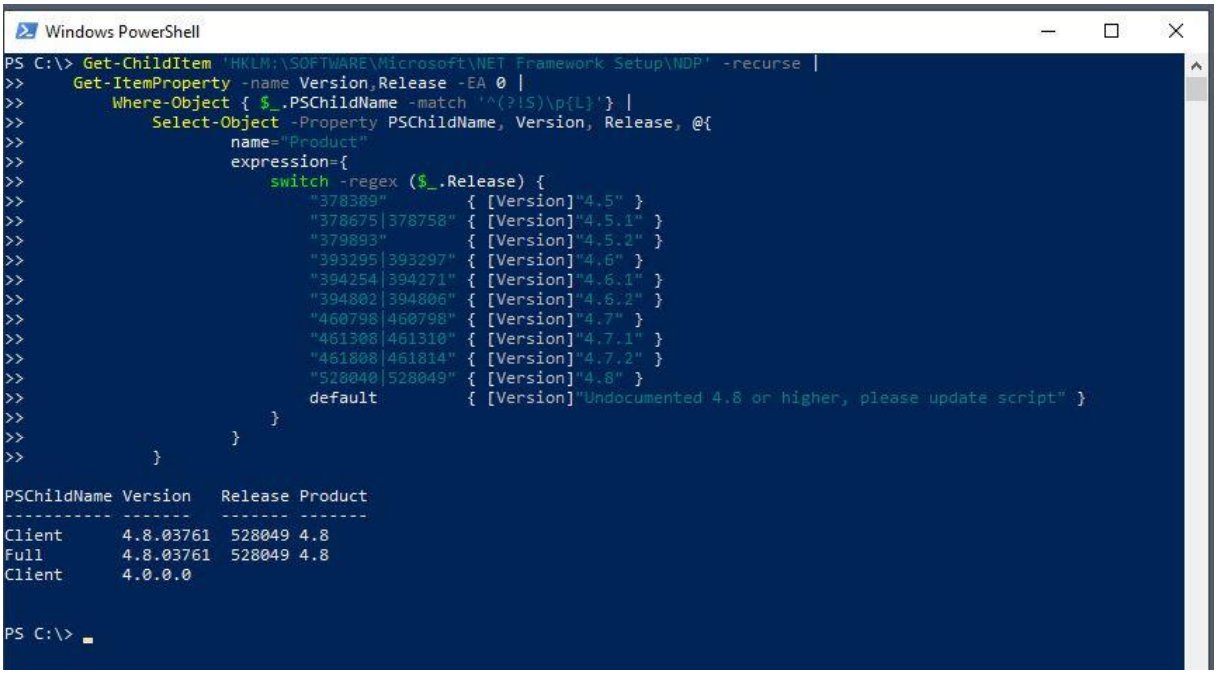


Mit einem Offline-Installer ist das aber schnell nachgeholt:



Nach dem Setup prüfe ich wieder mit Windows Updates auf Aktualisierungen. Natürlich wird auch hier etwas gefunden:

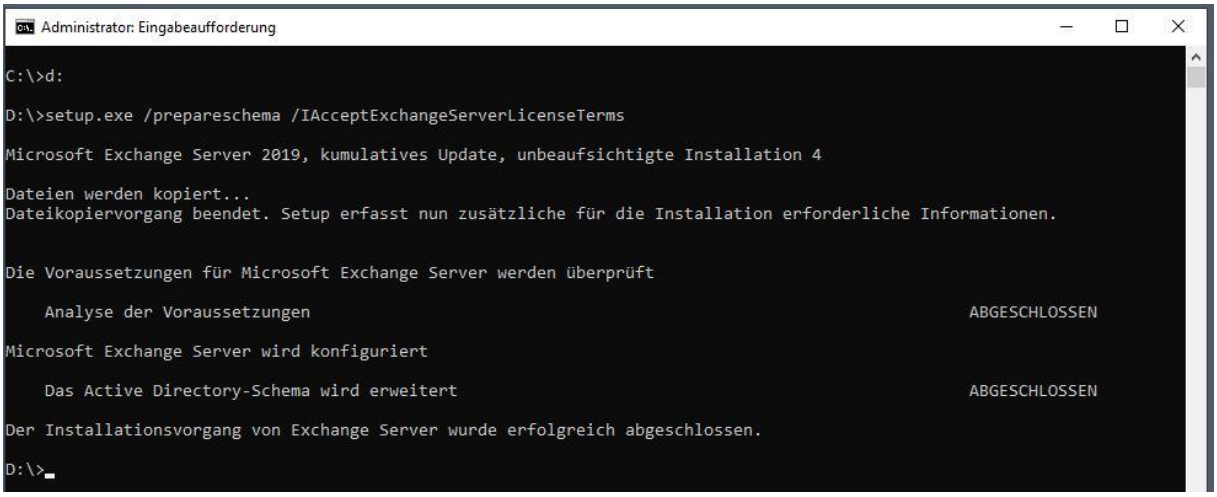
Das Setup und das Update beende ich mit einem Neustart. Jetzt passt die installierte Version:



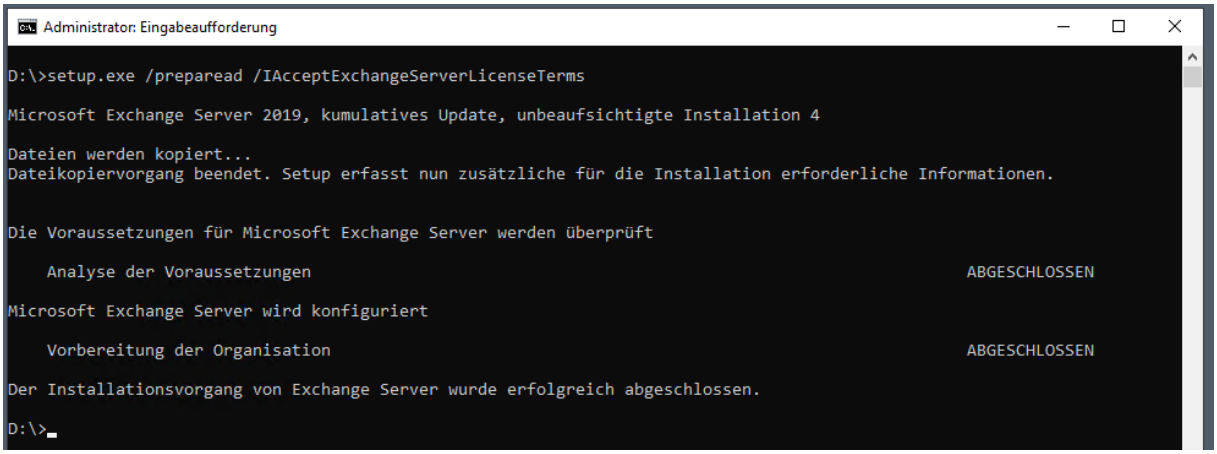
Ich möchte die Vorbereitung des Active Directory losgelöst vom Exchange Server Setup durchführen. Das bietet sich immer an, wenn man mehr als einen Domain Controller verwendet. Anderenfalls könnte die Aktualisierung auf einem DC1 angewendet werden, das Setup aber mit dem DC2 fortfahren. Die Replikation der beiden Domain Controller ist aber keine Echtzeit. Und gerade nach Schema-Veränderungen wird sie zusätzlich verzögert. Das Setup des neuen Exchange Servers kann so in einen Fehler laufen. Das will ich gerne vermeiden. Daher starte ich die Aktualisierung meiner Domain und warte dann auf die Replikation der Domain Controller.

Die Aktualisierung wird mit dem Installations-ISO und dem Aufruf der darin enthaltenen setup.exe durchgeführt:

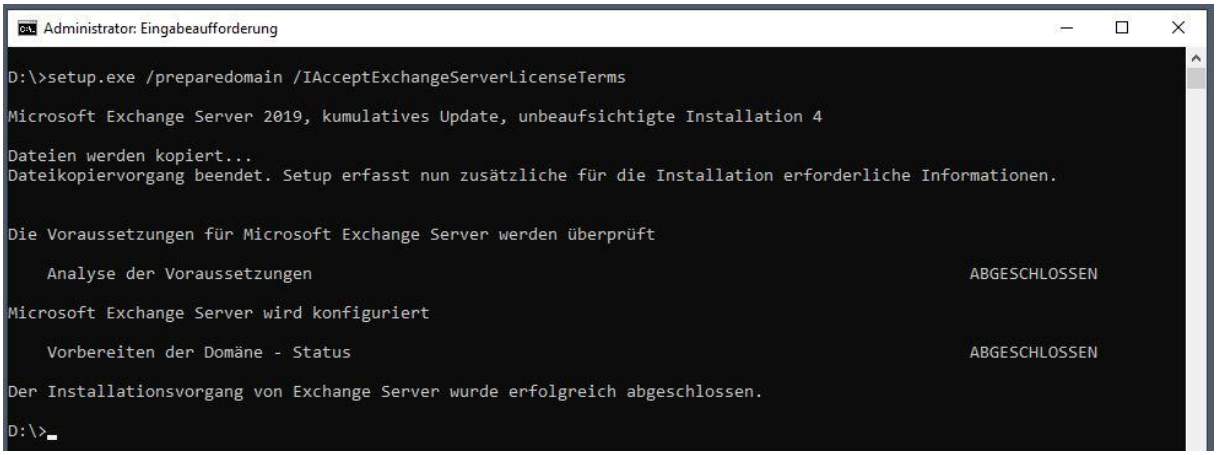
```
setup.exe /prepareschema /IAcceptExchangeServerLicenseTerms
```



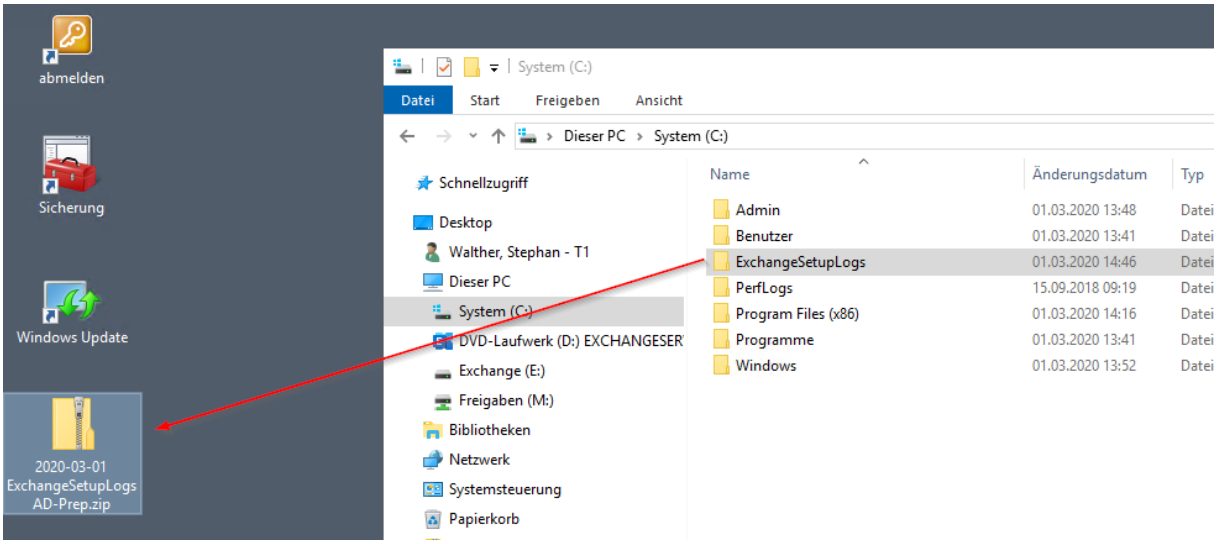
Der nächste Befehl aktualisiert die Configuration-Partition der Gesamtstruktur:



Abschließend starte ich noch die Aktualisierung der Domain Partition. Da ich nur eine Domain im meinem Active Directory Forest verwende muss ich den Befehl auch nur einmal starten:

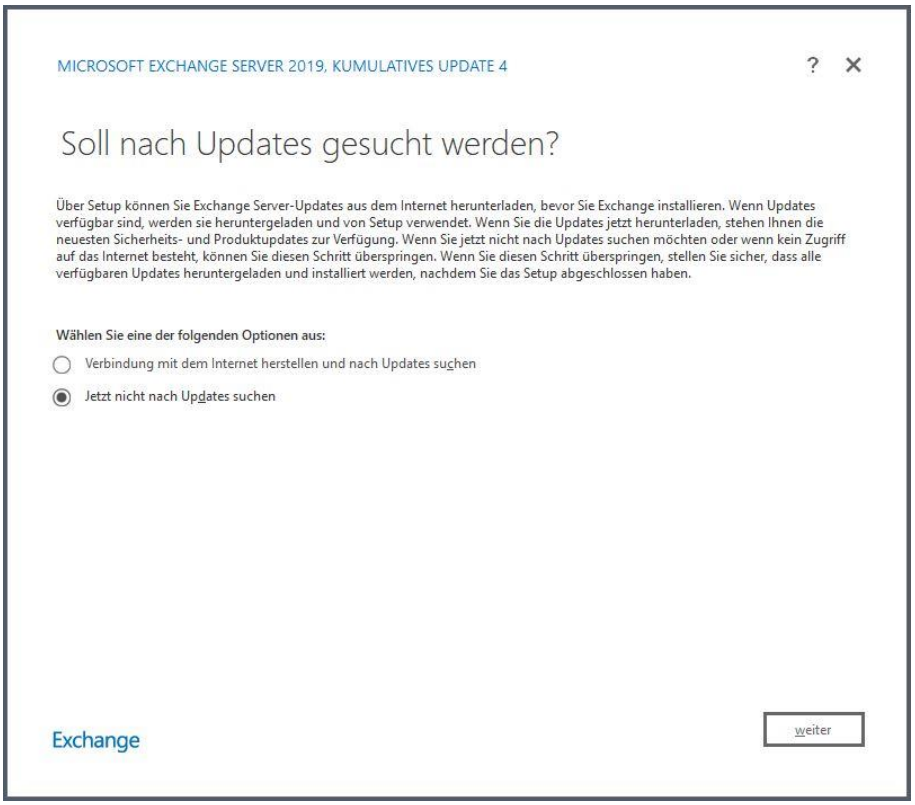


Nach wenigen Minuten ist alles erledigt. Die beim Aktualisieren geschriebenen Logfiles im Verzeichnis c:\ExchangeSetupLogs archiviere ich im Admin-Share. Logfiles von relevanten Veränderungen sind später immer wertvoll bei der Fehlersuche:

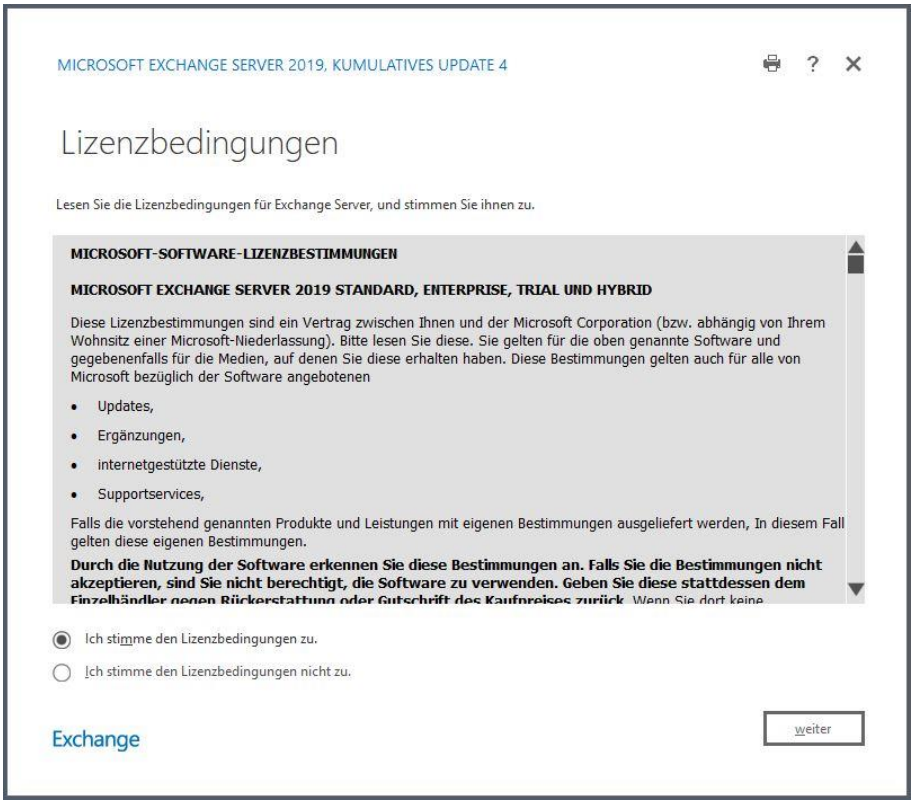


Installation des Exchange Servers 2019 CU4

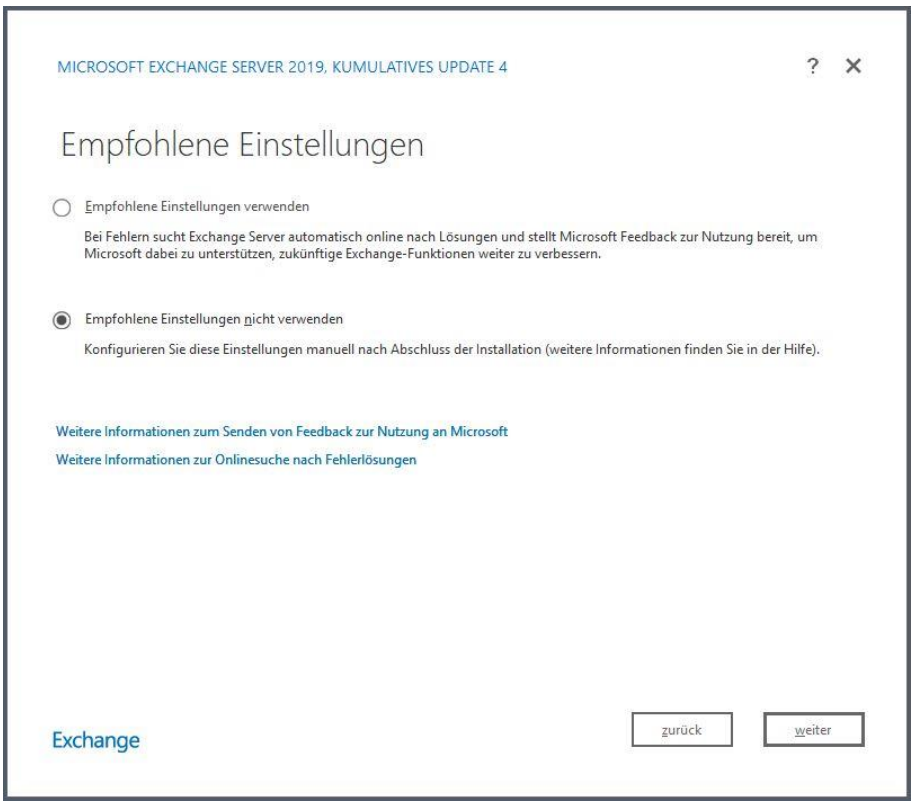
Die Installation des ersten Exchange Server 2019 benötigt noch einige Voraussetzungen. Diese werden aber vom Setup entweder auf Wunsch mitinstalliert oder als fehlend bei der Vorprüfung angegeben. Ich starte das Setup auf dem Server WS-MX2. Eine Update-Suche ist nicht möglich, da der neue Server nicht ins Internet kommt:



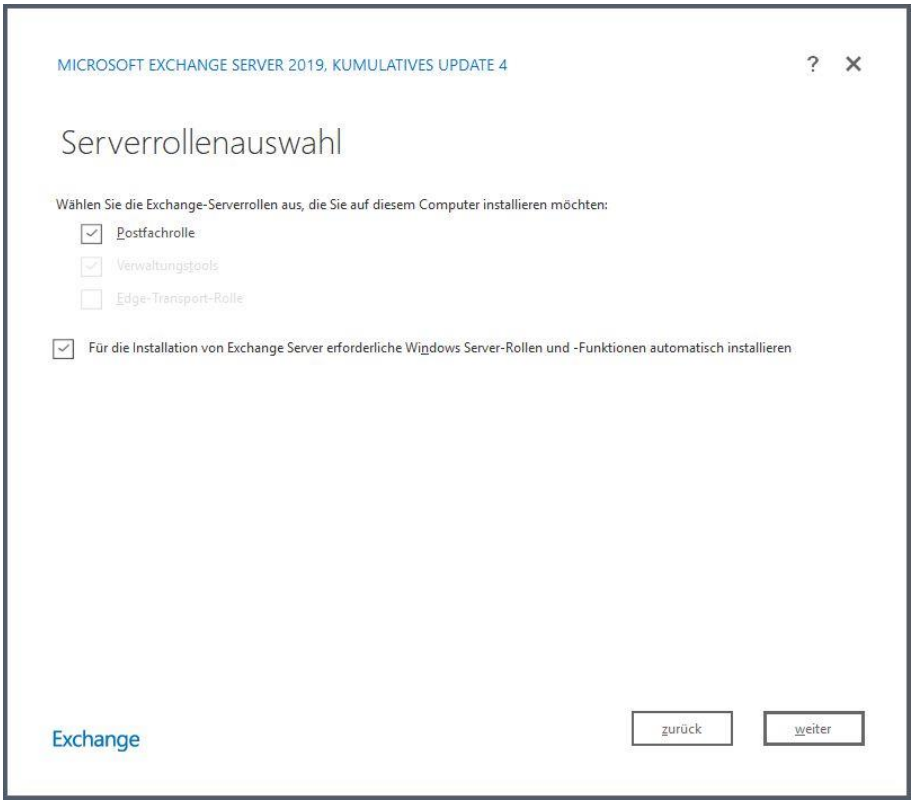
Nach dem ausführlichen Lesen der EULA bestätige ich diese:



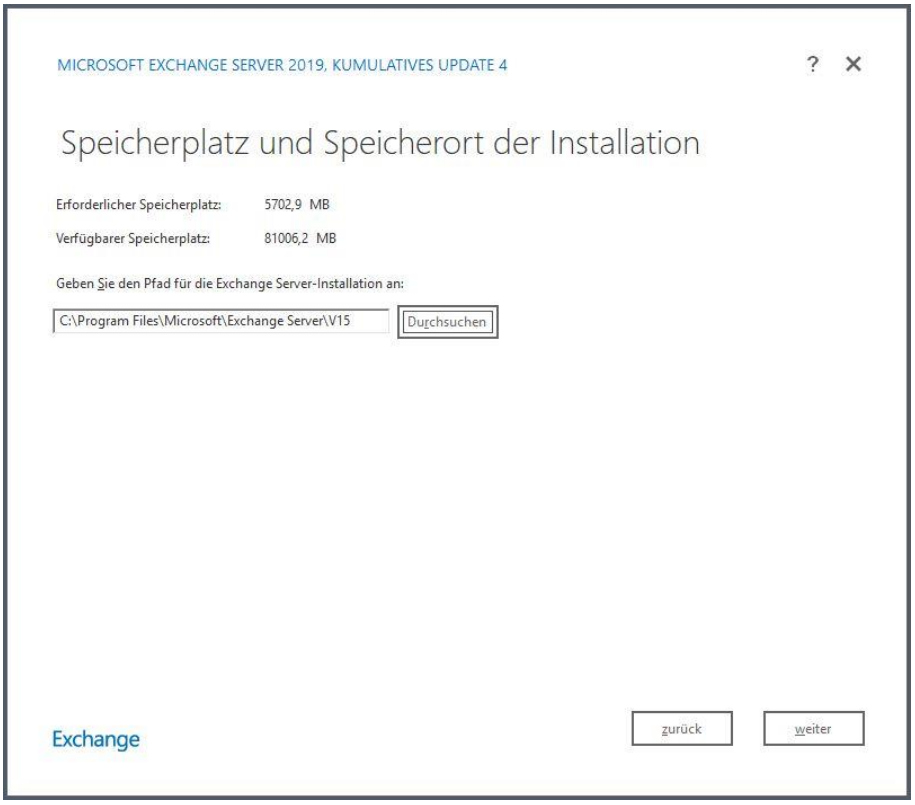
Ich möchte gerne die volle Kontrolle über den Installationsprozess. Daher wähle ich diese Option aus:



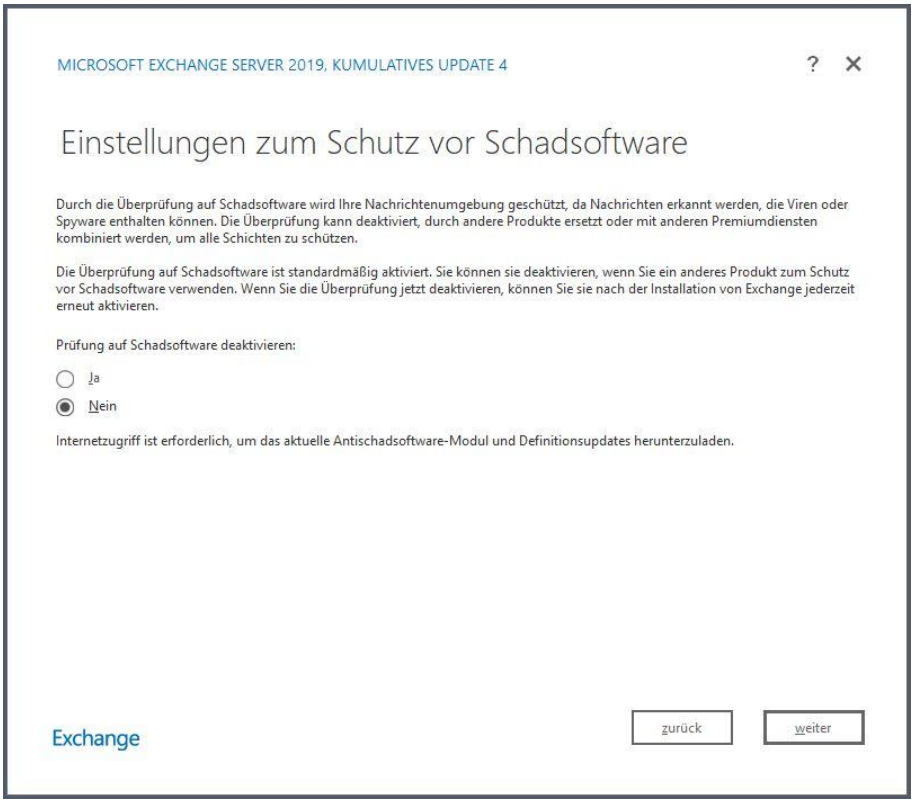
Im nächsten Fenster wähle ich die automatische Installation fehlender Komponenten aus. Bei den Rollen enthält die Rolle Mailbox seit Exchange Server 2016 die Datenbankrolle, den ClientAccess, das Unified Messaging und den Hubtransport-Service. Der Edge-Transport-Service ist wie in den Vorgängerversionen auch ein vorgelagerter SMTP-Server für die DMZ. Beide Rollen schließen sich gegenseitig bei der Installation aus:



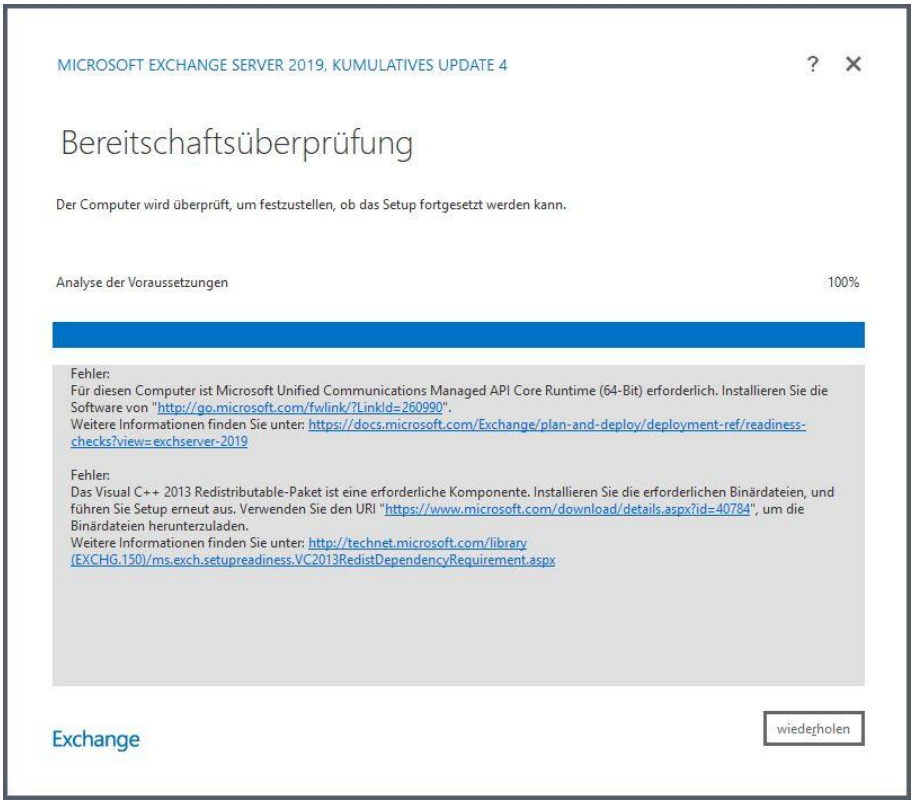
Die Installation soll im Standardverzeichnis landen. So kann ich mit einem SystemImage und der Sicherung der Systempartition das Betriebssystem und den Exchange Server als eine Einheit sichern und wiederherstellen:



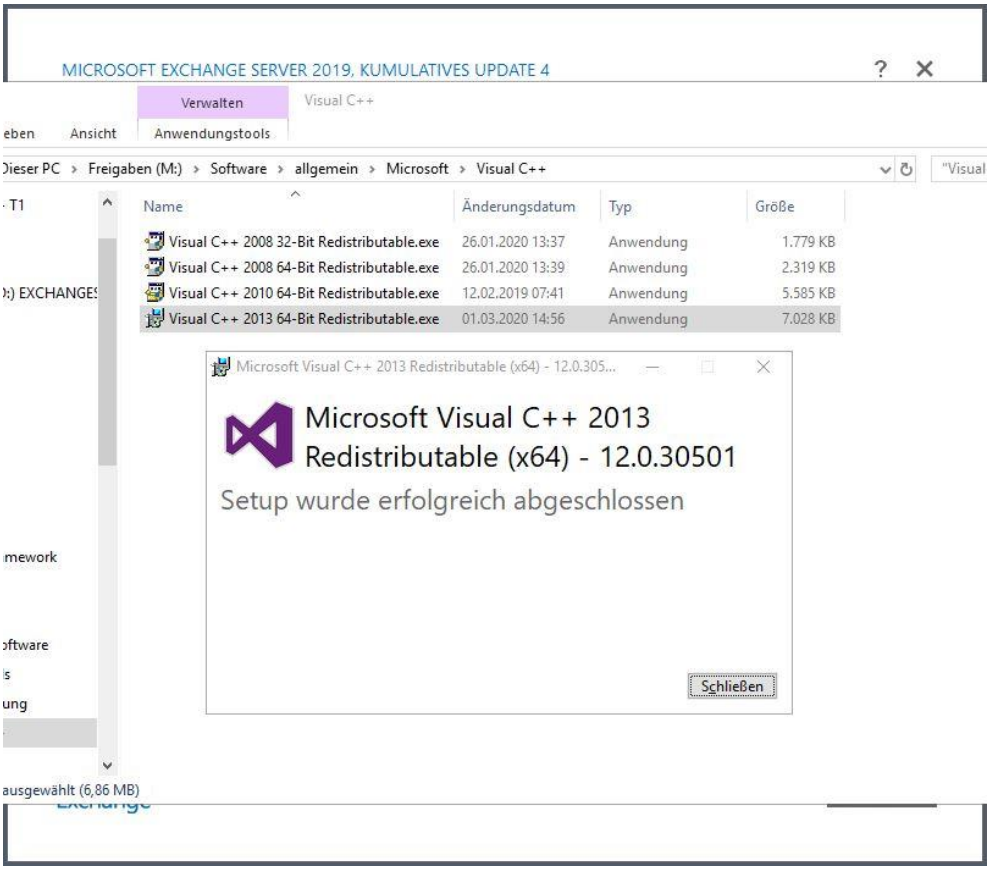
Die Prüfung auf Schadsoftware muss später manuell konfiguriert werden. Daher belasse ich die Einstellung unverändert:

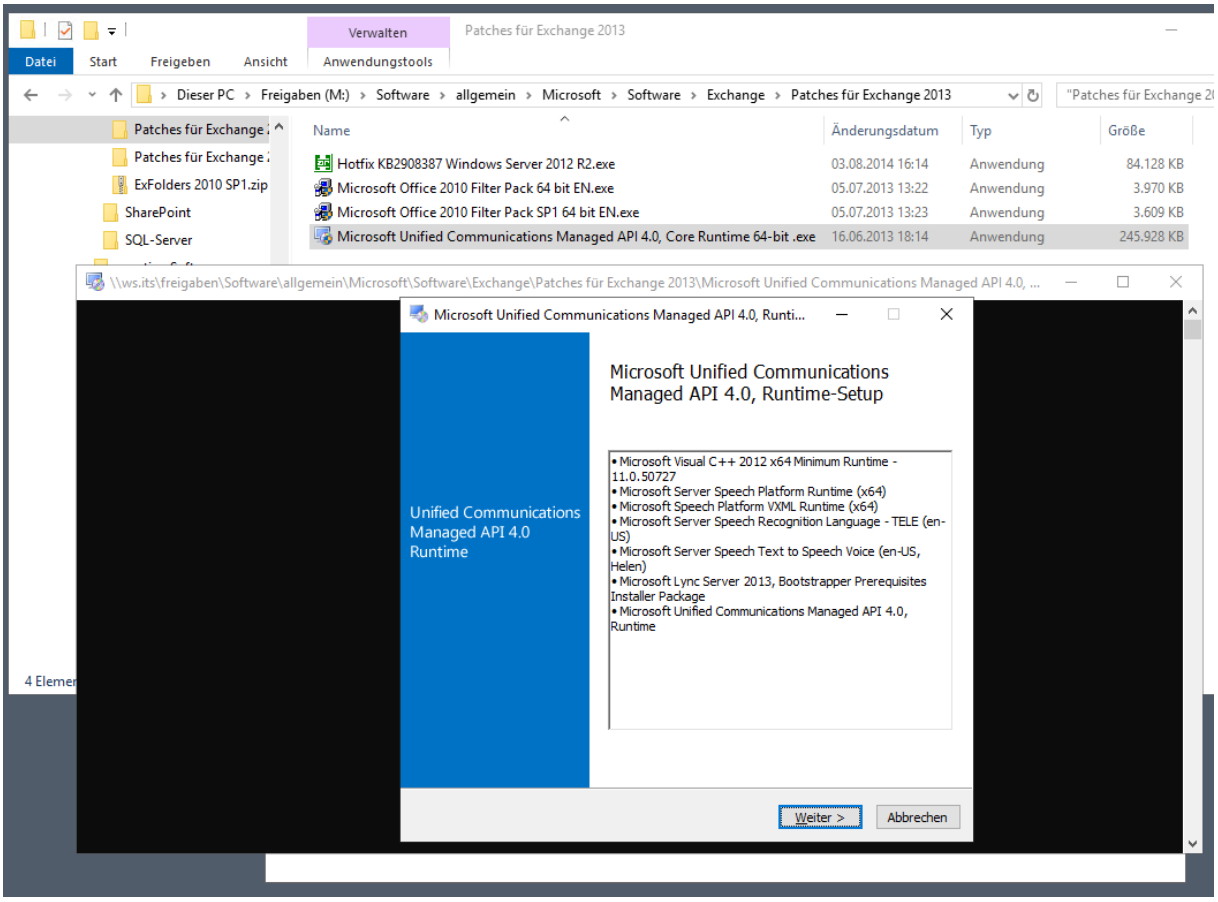


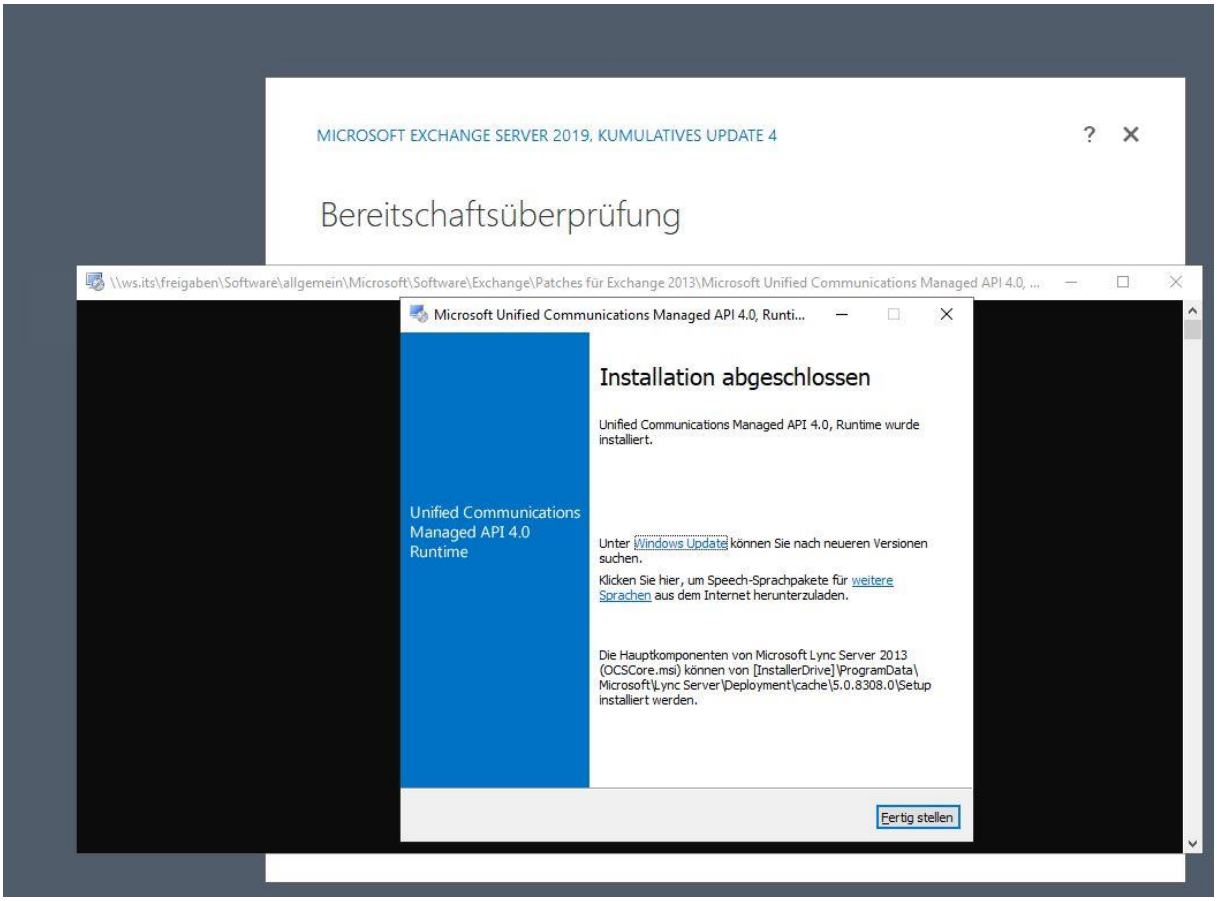
Das Setup installiert die erforderlichen Rollen und Features und bleibt mit einer Auflistung fehlender Voraussetzungen stehen:



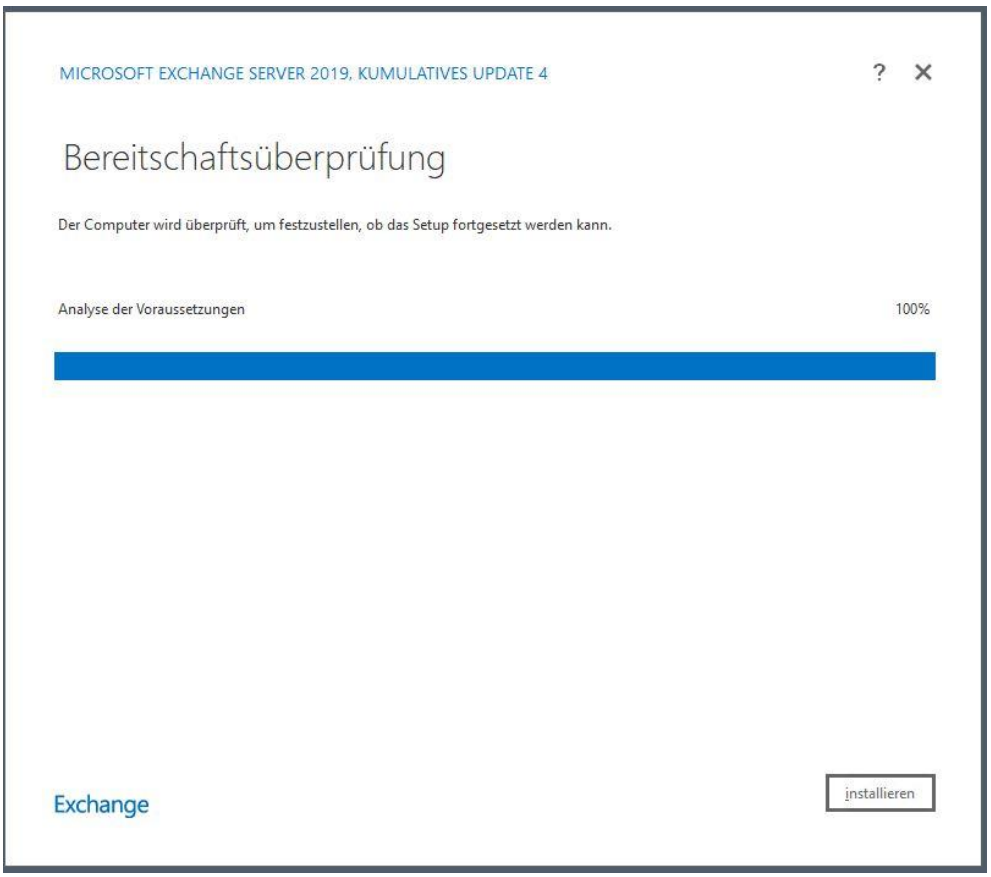
Beide Komponenten sind kein Problem. Sie können kostenlos bei Microsoft heruntergeladen werden. Ich habe beide bereits im Software-Share auf meinem Fileserver vorrätig:


MICROSOFT EXCHANGE SERVER 2019, KUMULATIVES UPDATE 4
Verwalten
Visual C++
Ansicht
Anwendungstools
Dieser PC › Freigaben (M:) › Software › allgemein › Microsoft › Visual C++
Name
Änderungsdatum
Typ
Größe
Visual C++ 2008 32-Bit Redistributable.exe 26.01.2020 13:37 Anwendung 1.779 KB
Visual C++ 2008 64-Bit Redistributable.exe 26.01.2020 13:39 Anwendung 2.319 KB
Visual C++ 2010 64-Bit Redistributable.exe 12.02.2019 07:41 Anwendung 5.585 KB
Visual C++ 2013 64-Bit Redistributable.exe 01.03.2020 14:56 Anwendung 7.028 KB
Microsoft Visual C++ 2013 Redistributable (x64) - 12.0.305...
Microsoft Visual C++ 2013 Redistributable (x64) - 12.0.30501
Setup wurde erfolgreich abgeschlossen
Schließen
ausgewählt (6,86 MB)

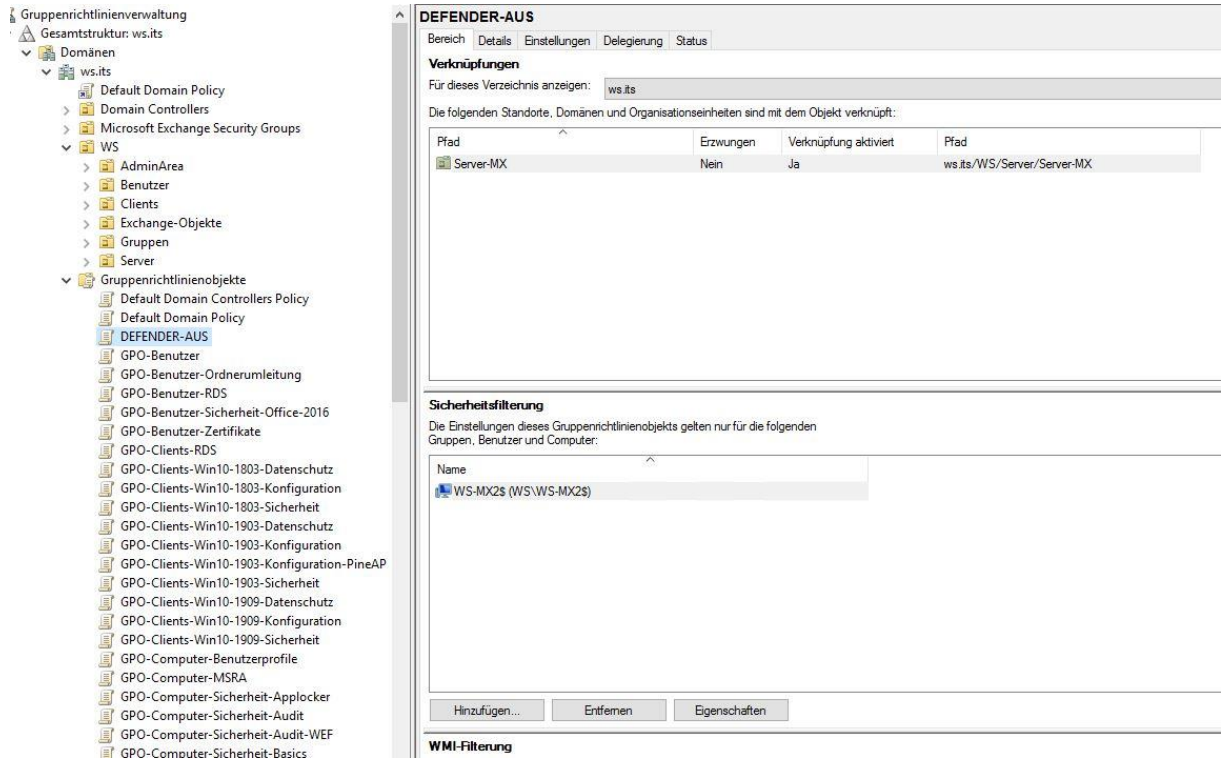

Verwalten
Patches für Exchange 2013
Datei
Start
Freigeben
Ansicht
Anwendungstools
Dieser PC › Freigaben (M:) › Software › allgemein › Microsoft › Software › Exchange › Patches für Exchange 2013
Name
Änderungsdatum
Typ
Größe
Hotfix KB2908387 Windows Server 2012 R2.exe 03.08.2014 16:14 Anwendung 84.128 KB
Microsoft Office 2010 Filter Pack 64 bit EN.exe 05.07.2013 13:22 Anwendung 3.970 KB
Microsoft Office 2010 Filter Pack SP1 64 bit EN.exe 05.07.2013 13:23 Anwendung 3.609 KB
Microsoft Unified Communications Managed API 4.0, Core Runtime 64-bit .exe 16.06.2013 18:14 Anwendung 245.928 KB
ExFolders 2010 SP1.zip
SharePoint
SQL-Server
Microsoft Unified Communications Managed API 4.0, Runti...
Microsoft Unified Communications Managed API 4.0, Runtime-Setup
Unified Communications Managed API 4.0 Runtime
• Microsoft Visual C++ 2012 x64 Minimum Runtime - 11.0.50727
• Microsoft Server Speech Platform Runtime (x64)
• Microsoft Speech Platform VXML Runtime (x64)
• Microsoft Server Speech Recognition Language - TELE (en-US)
• Microsoft Server Speech Text to Speech Voice (en-US, Helen)
• Microsoft Lync Server 2013, Bootstrapper Prerequisites Installer Package
• Microsoft Unified Communications Managed API 4.0, Runtime
Weiter >
Abbrechen



Die Installationen waren wie immer sehr unspektakulär. Mit einer Wiederholung der Voraussetzungsprüfung kehre ich zum Setup des neuen Exchange Servers zurück. Dieses Mal sind alle Voraussetzungen erfüllt:



Bevor ich das Setup starte, deaktiviere ich den Windows Defender. Dieser ist sein Windows Server 2016 eine bereits installierte Schutzkomponente. Leider grätscht er gerne in das Setup rein. Daher beende ich ihn. Das geht bei mir nur mit einer Gruppenrichtlinie, da eine andere GPO den Defender aktiviert. Mit einem Sicherheitsfilter auf der GPO treffe ich nur den neuen Exchange Server:



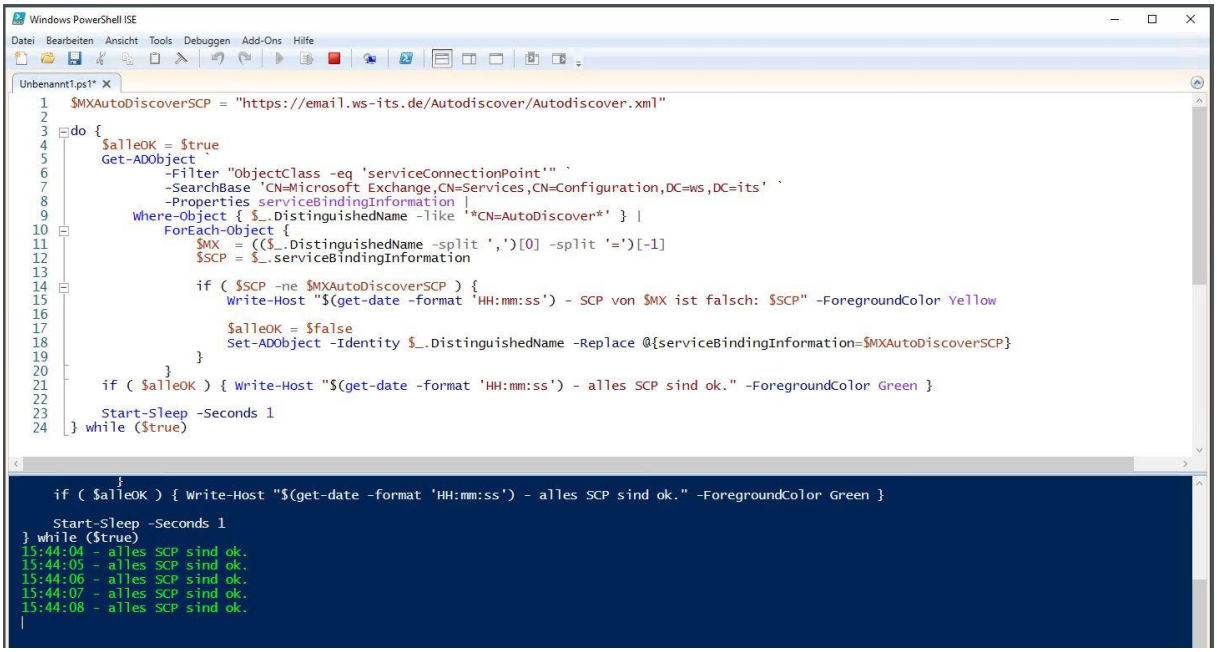
Zusätzlich starte ich auf einem anderen Server ein PowerShell-Script. Dieses scannt im Sekundentakt das Active Directory auf ServiceConnectionPoints (SCP). Hier trägt jeder Exchange Server seine URL ein, die von Outlook-Clients und ActiveSync-Clients zum Auffinden der Server abgefragt werden können. Der Default-Wert der URL enthält dabei immer der FQDN des Servers: https://<FQDN>/autodiscover/autodiscover.xml

Aber warum ist das problematisch und wird daher von mir bereits beim Setup korrigiert? Ganz einfach:

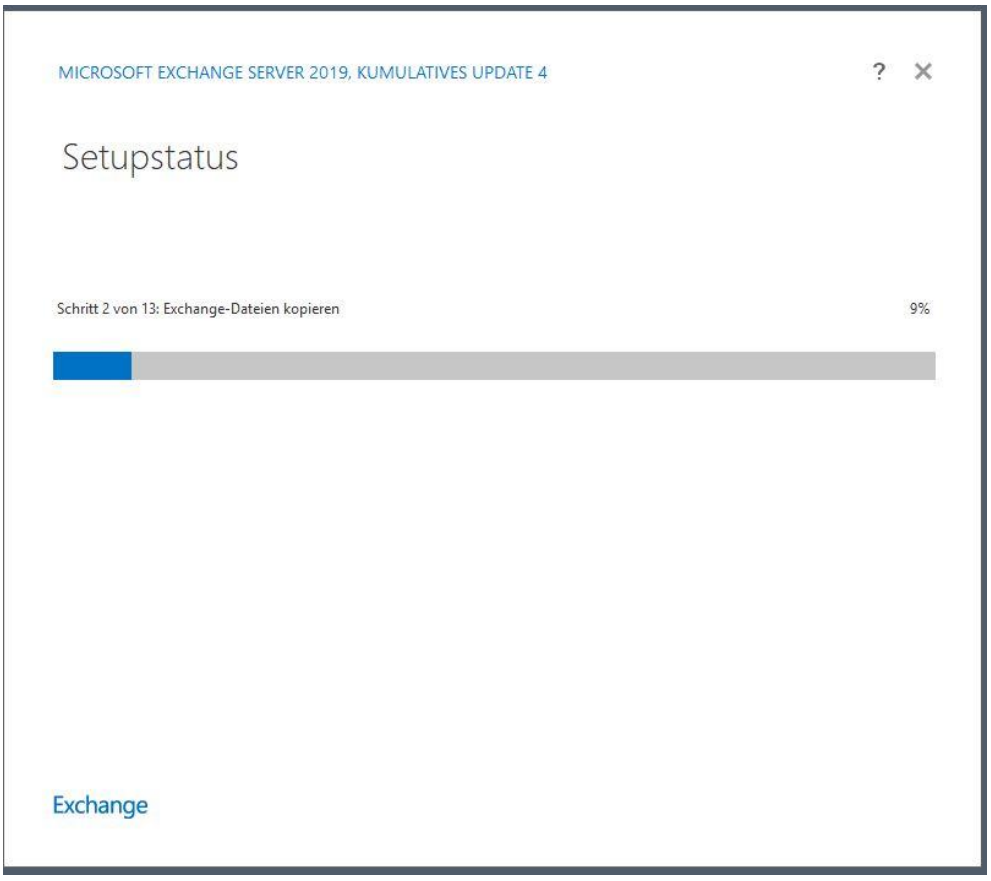
1. Der Eintrag wird während dem Setup automatisch mit dem FQDN des Servers erstellt.
2. Üblicherweise wird der Server erst nach dem Setup und einem Neustart konfiguriert. Dabei wird der Namespace - z.B. mail.ws-its.de – in der URL hinterlegt (https://mail.ws-its.de/autodiscover/autodiscover.xml) und es wird ein passendes Zertifikat installiert.
3. Zwischen Schritt 1 und Schritt 2 könnten Outlook-Clients auf die Idee kommen, die Autodiscover-Informationen zu aktualisieren. Dabei fragen sie einen Domain Controller nach SCP für Autodiscover. In diesem Fall würde ein DC neben der eigentlichen Namespace-URL https://mail.ws-its.de/autodiscover/autodiscover.xml auch die URL https://<FQDN>/autodiscover/autodiscover.xml in der Antwort versenden. Der Client hat also eine 50% Chance, dass er eine Verbindung zum neuen Server aufbaut.
4. Und je nach Setup-Stand kann dieser bereits aktiv sein. Leider hat er bis zur finalen Konfiguration im Schritt 2 ein selbstsigniertes Zertifikat, dem die Outlook-Clients natürlich nicht vertrauen. Nur leider haben sie dann schon das lokale Outlook-Profil verändert. Oft ist dieses irreparabel gestört und muss dann vom Helpdesk auf den Clients zurückgesetzt werden.

Da hilft auch kein Loadbalancer, der vor den Exchange Servern steht, denn die Clients umgehen diesen ja beim direkten Verbindungsaufbau mit dem Exchange Server! Bei mir würde das eine Firewall zwischen dem Servernetz und den Client-Netzen verhindern. Aber bei Kunden hatte ich dieses Problem schon mehrfach beobachtet.

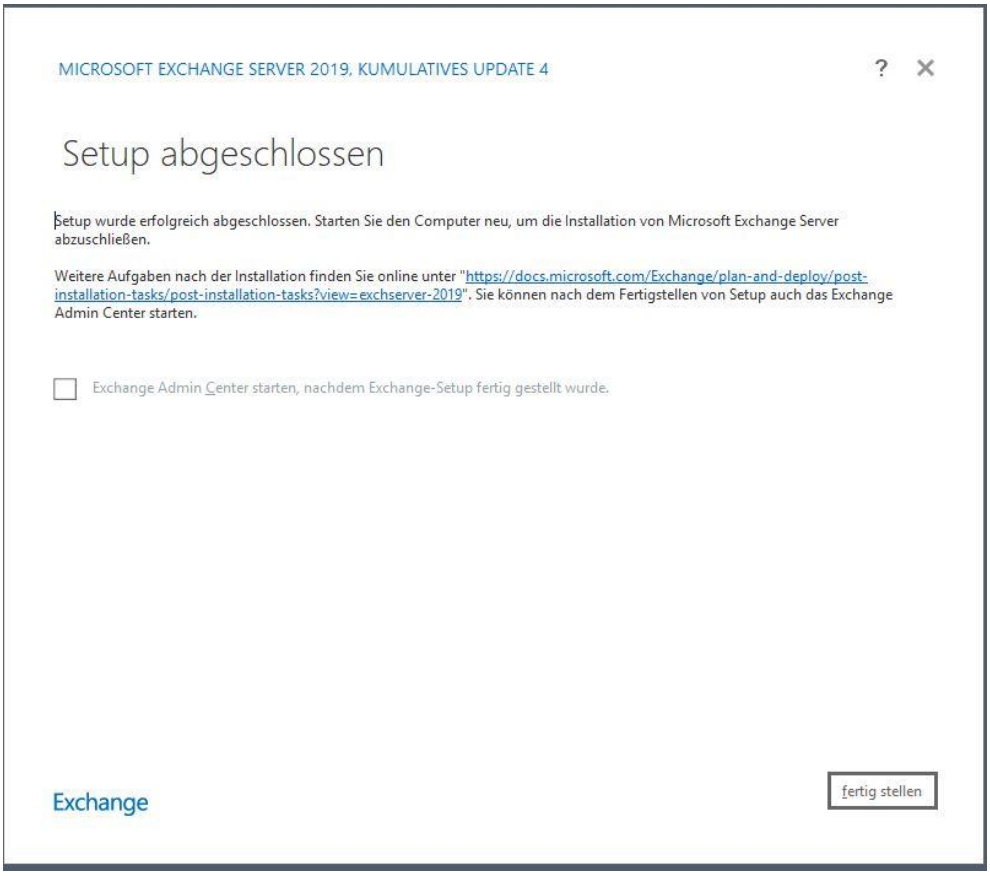
Warum das Microsoft nicht selber löst, indem der Record vielleicht zu Begin leer bleibt, weiß ich nicht. Aber mein Script kann die Korrektur direkt erkennen und den richtigen URL-Eintrag einschreiben. So bleibt für das oben genannte Szenario nahezu keine Zeit und es werden im Idealfall keine Outlook-Profile zerstört. Ich starte das Script auf einem anderen Server:



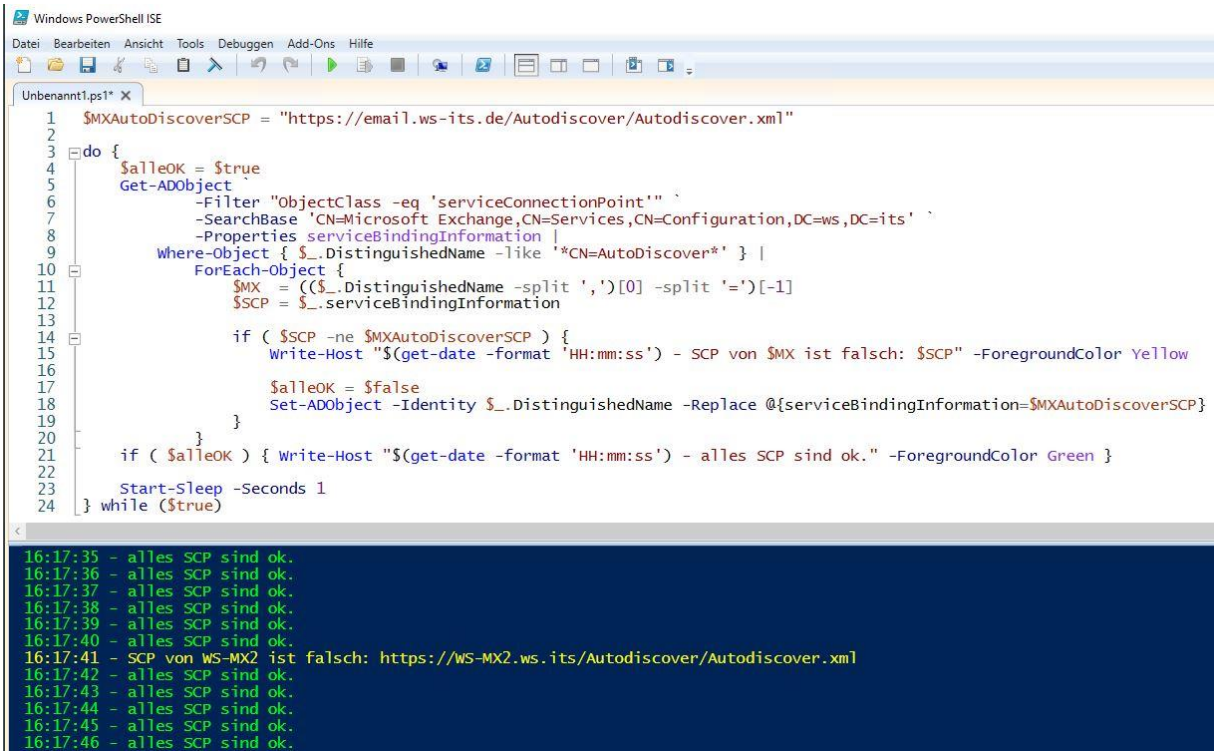
Jetzt sind alle Komponenten bereitgestellt. Ich starte das immer noch wartende Setup:



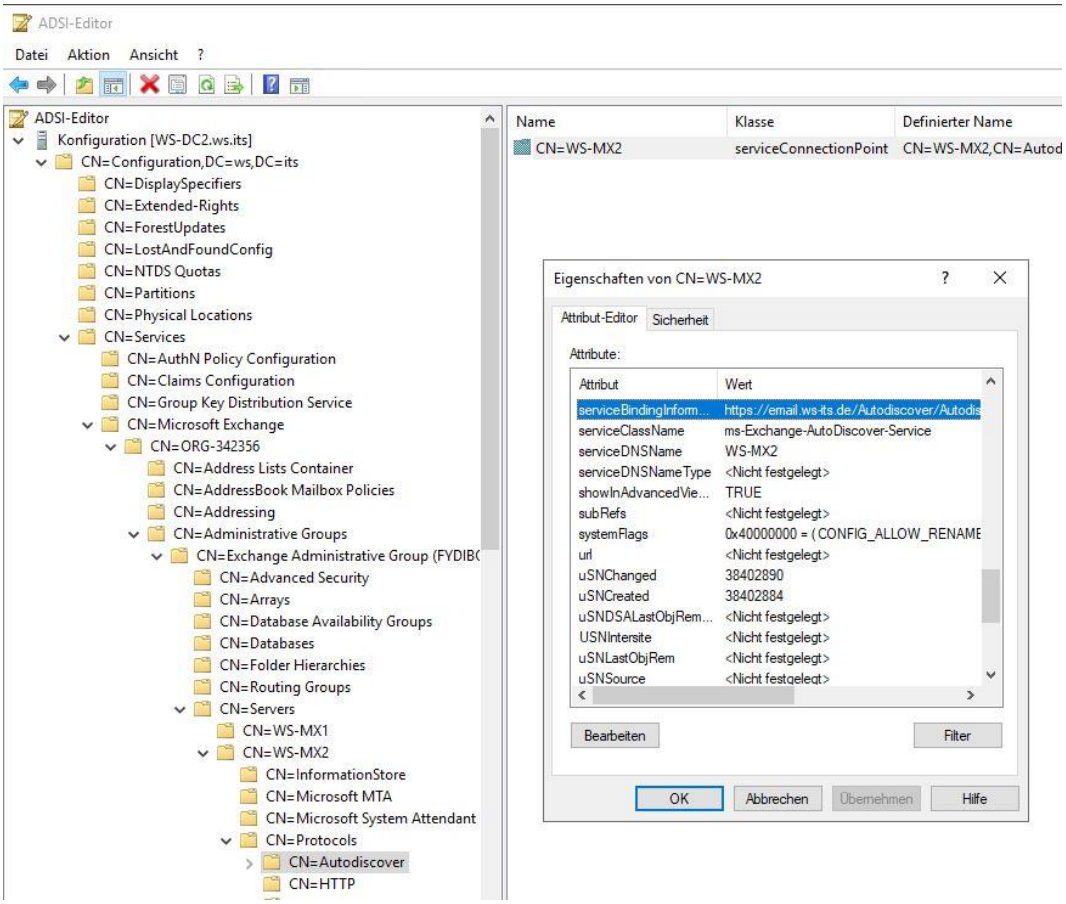
Einige Minuten später ist es erfolgreich abgeschlossen und muss mit einem Neustart finalisiert werden:



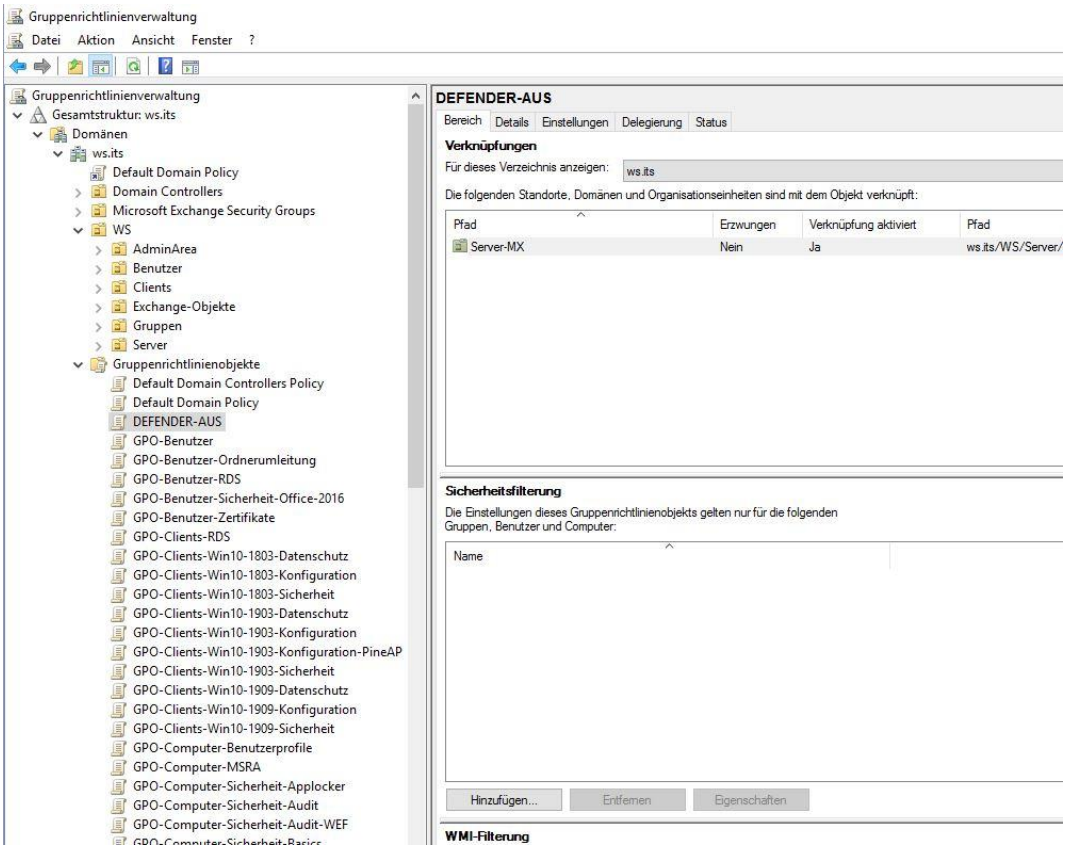
Auf dem anderen Server kontrolliere ich mein Script zur SCP-Korrektur. Im Logging finde ich den Zeitpunkt mit dem falschen Record. Und eine Sekunde später war er gefixt:



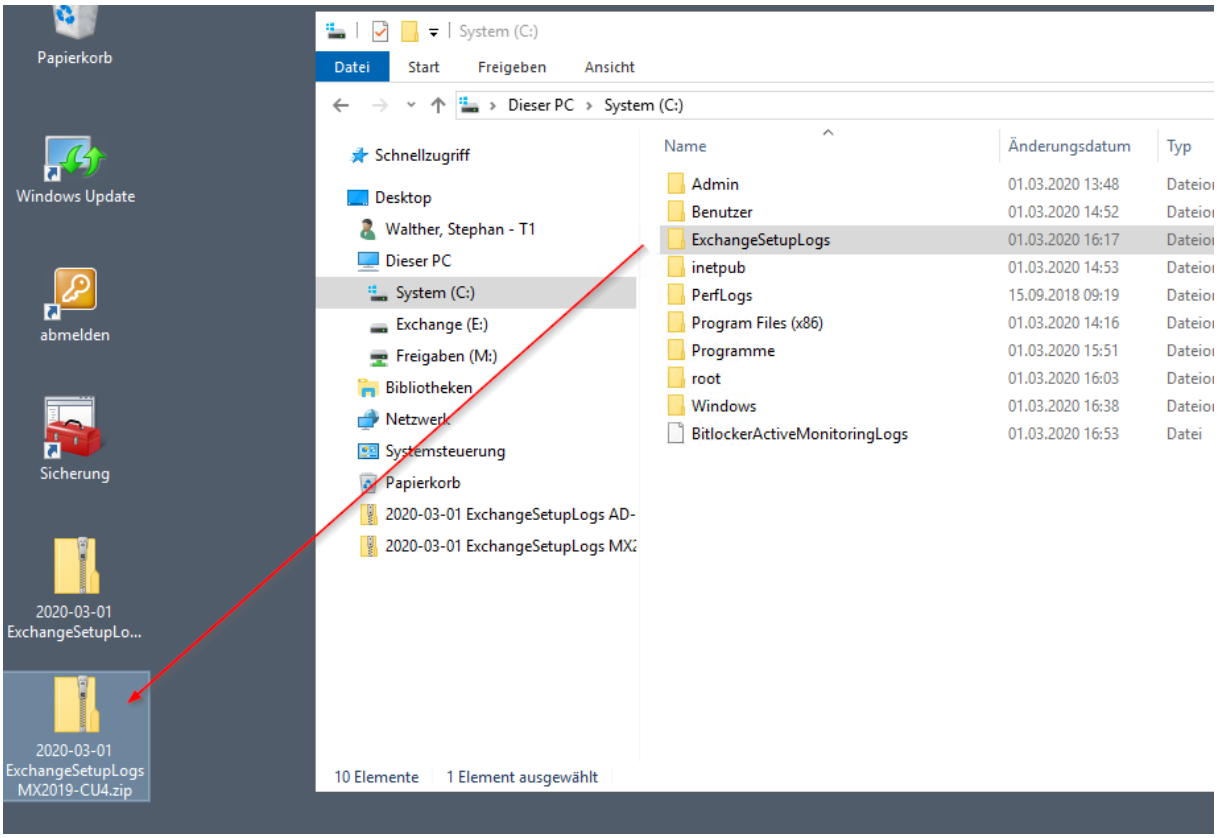
Im Active Directory kann man mit ADSIEdit den Record natürlich auch sehen:



Jetzt wird es auch wieder Zeit für den Defender. Ich entferne den Sicherheitsfilter der GPO und aktualisiere die Gruppenrichtlinien auf dem Server WS-MX2:



Die vom Setup geschriebenen Logfiles archiviere ich wieder in meinem Admin-Share:



Das Setup ist jetzt abgeschlossen. Jetzt folgt die Konfiguration der für mich relevanten Exchange Server Rollen.

Konfiguration der CAS-Rolle

Konfiguration der Virtual Directories

Die erste Rolle ist der ClientAccessService. Über diese Webdienste greifen alle Clients auf ihre Mailboxen zu. Dafür muss ich den Namespace „email.ws-its.de“ in alle virtuellen Verzeichnisse des Internet Information Services eintragen. Das geht sehr einfach mit der PowerShell. Wichtig an dieser Stelle wäre noch der Hinweis, dass ich alle Befehle von meinem Admin-Server ausführe. Die Anmeldung auf dem Exchange Server vermeide ich nach Möglichkeit.

Zuerst stelle ich in der PowerShell-ISE eine Verbindung zum Exchange Server her:

```
PS C:\> $mx = New-PSSession -ConfigurationName microsoft.exchange -ConnectionUri http://ws-mx2.ws.its/powershell -Authentication kerberos
Import-PSSession $mx -DisableNameChecking

ModuleType Version    Name                                ExportedCommands
---------- -------    ----                                ----------------
Script     1.0        tmp_egs0fvl4.x32                    {Add-ADPermission, Add-AvailabilityAddressSpace, Add-ContentFilterPhrase, Add-Databas...
```

Intern wie extern verwende ich den gleichen Namespace. Die DNS-Server liefern dazu je nach Netzwerk die interne oder die externe IPv4-Adresse des LoadBalancers. Zeile 69 ändert übrigens den URL-Eintrag im Active Directory für das Autodiscover – der ServiceConnectionPoint, den mein Script vorhin schon veränderte:

```powershell
# Konfiguration der Rolle CAS
    $servername = "WS-MX2"
    $internalhostname     = "email.ws-its.de"
    $externalhostname     = "email.ws-its.de"
    $autodiscoverhostname = "email.ws-its.de"

    $owainturl  = "https://$internalhostname/owa"
    $owaexturl  = "https://$externalhostname/owa"
    $ecpinturl  = "https://$internalhostname/ecp"
    $ecpexturl  = "https://$externalhostname/ecp"
    $ewsinturl  = "https://$internalhostname/EWS/Exchange.asmx"
    $ewsexturl  = "https://$externalhostname/EWS/Exchange.asmx"
    $easinturl  = "https://$internalhostname/Microsoft-Server-ActiveSync"
    $easexturl  = "https://$externalhostname/Microsoft-Server-ActiveSync"
    $oabinturl  = "https://$internalhostname/OAB"
    $oabexturl  = "https://$externalhostname/OAB"
    $mapiinturl = "https://$internalhostname/mapi"
    $mapiexturl = "https://$externalhostname/mapi"
    $aduri      = "https://$autodiscoverhostname/Autodiscover/Autodiscover.xml"

    Get-OwaVirtualDirectory         -Server $servername | Set-OwaVirtualDirectory         -internalurl $owainturl  -externalurl $owaexturl
    Get-EcpVirtualDirectory         -Server $servername | Set-EcpVirtualDirectory         -internalurl $ecpinturl  -externalurl $ecpexturl
    Get-WebServicesVirtualDirectory -Server $servername | Set-WebServicesVirtualDirectory -internalurl $ewsinturl  -externalurl $ewsexturl
    Get-ActiveSyncVirtualDirectory  -Server $servername | Set-ActiveSyncVirtualDirectory  -internalurl $easinturl  -externalurl $easexturl
    Get-OabVirtualDirectory         -Server $servername | Set-OabVirtualDirectory         -internalurl $oabinturl  -externalurl $oabexturl
    Get-MapiVirtualDirectory        -Server $servername | Set-MapiVirtualDirectory        -internalurl $mapiinturl -externalurl $mapiexturl
    Get-OutlookAnywhere             -Server $servername | Set-OutlookAnywhere `
            -externalhostname $externalhostname `
            -internalhostname $internalhostname `
            -ExternalClientsRequireSsl:$true `
            -InternalClientsRequireSsl:$true `
            -ExternalClientAuthenticationMethod 'Negotiate'

    Get-ClientAccessService       -Identity $servername | Set-ClientAccessService        -AutoDiscoverServiceInternalUri $aduri
    Get-ClientAccessService $servername
```

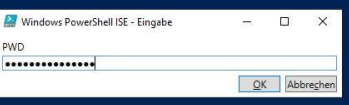
Die Zeilen laufen anstandslos durch.

Installation des Serverzertifikates

Aber ohne ein passendes und vor allem vertrauenswürdigem Serverzertifikat wird kein Client gerne eine Verbindung aufbauen wollen. Das bisherige Zertifikat ist noch ein paar Monate gültig. Es wurde von einer öffentlichen Zertifizierungsstelle digital signiert. Die PKCS12-Datei mit dem öffentlichen und dem dazugehörigen privaten Schlüssel liegt mit einem Passwort geschützt in meinem Admin-Share. Mit der PowerShell kopiere ich die Datei auf den RemoteServer und installiere dort das Zertifikat. So umgehe ich den DoppelHop (AdminServer ➔ Exchange Server ➔ FileServer)

```powershell
    # Installation des Serverzertifikates
        $servername = "WS-MX2"

        Copy-Item `
                -Path        "M:\AdminArea\Services\Exchange\Zertifikate\MX-2019-10-20-extern.pfx" `
                -Destination "\\$servername\c$\admin\cert.pfx"

        Invoke-Command -ComputerName $servername -ScriptBlock {
            Import-PfxCertificate `
                    -FilePath 'C:\Admin\cert.pfx' `
                    -Password (Read-Host -Prompt 'PWD' -AsSecureString) `
                    -CertStoreLocation  Cert:\LocalMachine\My
        }
# Konfiguration der Rolle HTS

# Konfiguration der Rolle MBS
```

```
PS C:\>   $servername = "WS-MX2"

        Copy-Item `
                -Path        "M:\AdminArea\Services\Exchange\Zertifikate\MX-2019-10-20-extern.pfx" `
                -Destination "\\$servername\c$\admin\cert.pfx"


PS C:\> Invoke-Command -ComputerName $servername -ScriptBlock {
            Import-PfxCertificate `
                    -FilePath 'C:\Admin\cert.pfx' `
                    -Password (Read-Host -Prompt 'PWD' -AsSecureString) `
                    -CertStoreLocation  Cert:\LocalMachine\My
        }
WARNUNG: Ein Skript oder eine Anwendung auf dem Remotecomputer "WS-MX2" sendet eine Eingabeaufforderung. Geben Sie bei Aufforderung nur dann vertra
uliche Informationen wie Anmeldeinformationen oder Kennwörter ein, wenn Sie dem Remotecomputer und der anfordernden Anwendung bzw. dem anfordernden
 Skript vertrauen.
```

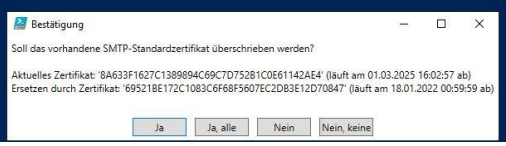



```
   PSParentPath: Microsoft.PowerShell.Security\Certificate::LocalMachine\My

Thumbprint                                Subject                                              PSComputerName
----------                                -------                                              --------------
69521BE172C1083C6F68F5607EC2DB3E12D70847  CN=email.ws-its.de, OU=Domain Control Validated      WS-MX2
```

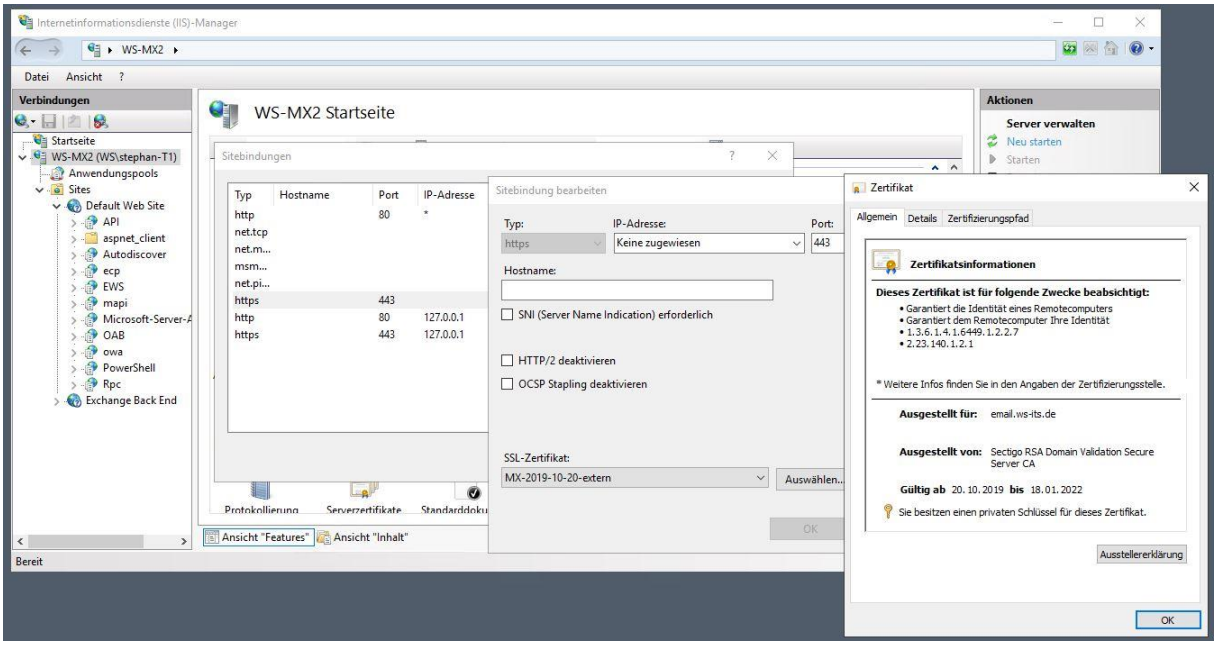
Jetzt kann ich das Zertifikat an die beiden Services SMTP und IIS binden. Die Warnmeldung zeigt die aktuelle Verwendung eines selbstsignierten Zertifikates an. Mit Freuden bestätige ich sie:

```powershell
    # Aktivierung des Serverzertifikates
        Enable-ExchangeCertificate -Thumbprint 69521BE172C1083C6F68F5607EC2DB3E12D70847 -Services SMTP -Server $servername
        Enable-ExchangeCertificate -Thumbprint 69521BE172C1083C6F68F5607EC2DB3E12D70847 -Services IIS  -Server $servername
```

```
PS C:\>         Enable-ExchangeCertificate -Thumbprint 69521BE172C1083C6F68F5607EC2DB3E12D70847 -Services SMTP -Server $servername
        Enable-ExchangeCertificate -Thumbprint 69521BE172C1083C6F68F5607EC2DB3E12D70847 -Services IIS  -Server $servername
```

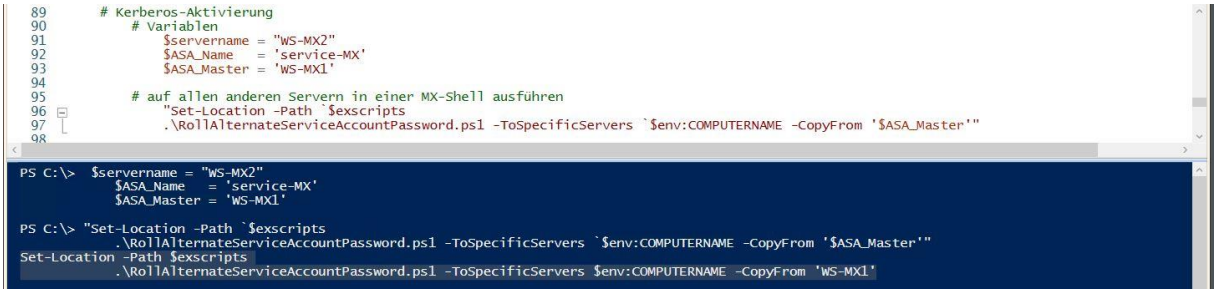


Eine kurze Kontrolle im Internet Information Service Manager (IIS-Manager) zeigt das eingebundene Zertifikat:



Umstellung auf Kerberos-Authentication

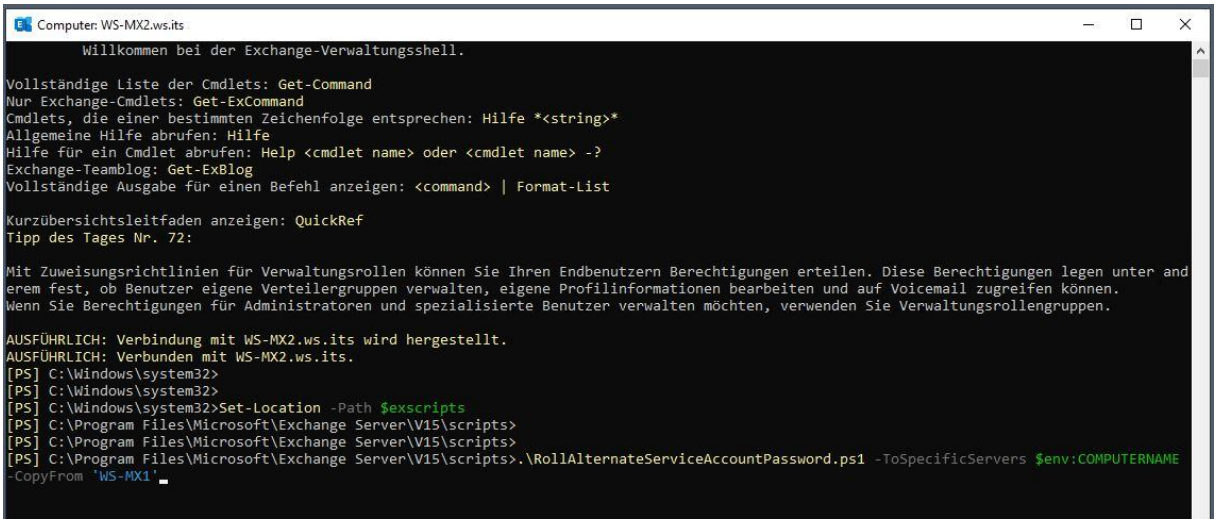
Vor einiger Zeit habe ich die Anmeldung meiner Clients am Exchange Service auf Kerberos umgestellt. Der neue Server muss dafür das Passwort des Accounts von einem anderen Server kopieren. Nur so können mehrere Exchange Server hinter einem LoadBalancer mit der gleichen Kerberos-Identität agieren. Der Quellserver ist der alte WS-MX1. Mit meinem Script in der PowerShell-ISE erzeuge ich nur die Befehle. Diese müssen in der Exchange Management Shell ausgeführt werden:



```
89      # Kerberos-Aktivierung
90          # Variablen
91              $servername = "WS-MX2"
92              $ASA_Name   = 'service-MX'
93              $ASA_Master = 'WS-MX1'
94
95          # auf allen anderen Servern in einer MX-Shell ausführen
96              "Set-Location -Path `$exscripts
97              .\RollAlternateServiceAccountPassword.ps1 -ToSpecificServers `$env:COMPUTERNAME -CopyFrom '$ASA_Master'"
98
```

```
PS C:\>  $servername = "WS-MX2"
         $ASA_Name   = 'service-MX'
         $ASA_Master = 'WS-MX1'

PS C:\> "Set-Location -Path `$exscripts
         .\RollAlternateServiceAccountPassword.ps1 -ToSpecificServers `$env:COMPUTERNAME -CopyFrom '$ASA_Master'"
Set-Location -Path $exscripts
         .\RollAlternateServiceAccountPassword.ps1 -ToSpecificServers $env:COMPUTERNAME -CopyFrom 'WS-MX1'
```

Ich starte auf dem neuen Exchange Server die Management Shell und starte das Script mit den 2 Befehlen:



```
Computer: WS-MX2.ws.its
        Willkommen bei der Exchange-Verwaltungsshell.

Vollständige Liste der Cmdlets: Get-Command
Nur Exchange-Cmdlets: Get-ExCommand
Cmdlets, die einer bestimmten Zeichenfolge entsprechen: Hilfe *<string>*
Allgemeine Hilfe abrufen: Hilfe
Hilfe für ein Cmdlet abrufen: Help <cmdlet name> oder <cmdlet name> -?
Exchange-Teamblog: Get-ExBlog
Vollständige Ausgabe für einen Befehl anzeigen: <command> | Format-List

Kurzübersichtsleitfaden anzeigen: QuickRef
Tipp des Tages Nr. 72:

Mit Zuweisungsrichtlinien für Verwaltungsrollen können Sie Ihren Endbenutzern Berechtigungen erteilen. Diese Berechtigungen legen unter and
erem fest, ob Benutzer eigene Verteilergruppen verwalten, eigene Profilinformationen bearbeiten und auf Voicemail zugreifen können.
Wenn Sie Berechtigungen für Administratoren und spezialisierte Benutzer verwalten möchten, verwenden Sie Verwaltungsrollengruppen.

AUSFÜHRLICH: Verbindung mit WS-MX2.ws.its wird hergestellt.
AUSFÜHRLICH: Verbunden mit WS-MX2.ws.its.
[PS] C:\Windows\system32>
[PS] C:\Windows\system32>
[PS] C:\Windows\system32>Set-Location -Path $exscripts
[PS] C:\Program Files\Microsoft\Exchange Server\V15\scripts>
[PS] C:\Program Files\Microsoft\Exchange Server\V15\scripts>
[PS] C:\Program Files\Microsoft\Exchange Server\V15\scripts>.\RollAlternateServiceAccountPassword.ps1 -ToSpecificServers $env:COMPUTERNAME
-CopyFrom 'WS-MX1'
```

```
Computer: WS-MX2.ws.its

========== Starting at 03/02/2020 19:03:05 ==========
Destination servers that will be updated:

Name   PSComputerName
----   --------------
WS-MX2 ws-mx2.ws.its


Credentials that will be pushed to every server in the specified scope (recent first):

UserName                         Password
--------                         --------
WS\service-MX$ System.Security.SecureString
WS\service-MX$ System.Security.SecureString


Prior to pushing new credentials, all existing credentials will be removed from the destination servers.
Pushing credentials to server WS-MX2
Retrieving the current Alternate Service Account configuration from servers in scope
Alternate Service Account properties:

StructuralObjectClass QualifiedUserName Last Pwd Update     SPNs
--------------------- ----------------- ---------------     ----
computer              WS\service-MX$    24.07.2019 13:35:23 http/email.ws-its.de
                                                            http/email.ws-its.de


Per-server Alternate Service Account configuration as of the time of script completion:


   Array: {email.ws-its.de, email.ws-its.de}

Identity AlternateServiceAccountConfiguration
-------- ------------------------------------
WS-MX2   Zuletzt: 02.03.2020 19:03:20, WS\service-MX$
         Zuvor: 02.03.2020 19:03:20, WS\service-MX$


========== Finished at 03/02/2020 19:03:21 ==========

       THE SCRIPT HAS SUCCEEDED
[PS] C:\Program Files\Microsoft\Exchange Server\V15\scripts>
```

Das war recht einfach. Eine kleine Kontrolle in der PowerShell-ISE bestätigt den Import:

```
 99          # Prüfung:
100              Get-ClientAccessService -Identity $servername -IncludeAlternateServiceAccountCredentialStatus |
101                  Format-List Name, AlternateServiceAccountConfiguration
102

PS C:\> Get-ClientAccessService -Identity $servername -IncludeAlternateServiceAccountCredentialStatus |
            Format-List Name, AlternateServiceAccountConfiguration


Name                                 : WS-MX2
AlternateServiceAccountConfiguration : Zuletzt: 02.03.2020 19:03:20, WS\service-MX$
                                       Zuvor: 02.03.2020 19:03:20, WS\service-MX$
```

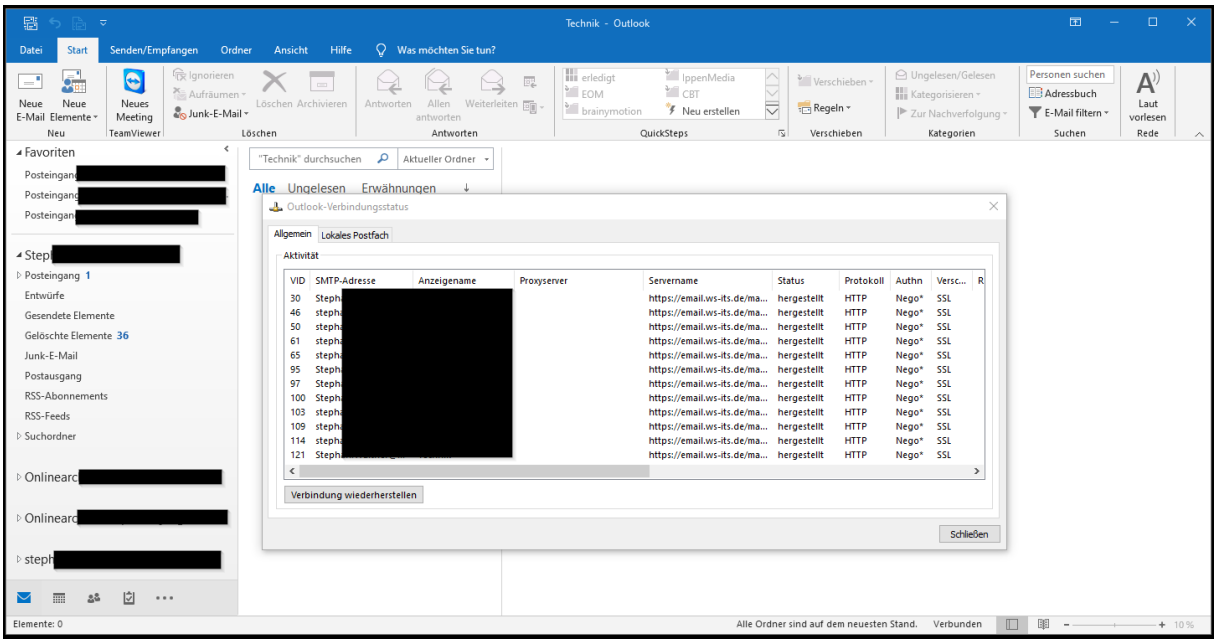
Jetzt rekonfiguriere ich noch die relevanten virtuellen Verzeichnisse. Das hätte ich natürlich auch vorher schon erledigen können. Aber jetzt passt es besser rein:

```
Windows PowerShell ISE
Datei  Bearbeiten  Ansicht  Tools  Debuggen  Add-Ons  Hilfe
MX-Migration.ps1*
102
103          # Aktivierung von Kerberos im CAS
104              Get-OutlookAnywhere -Server $servername | Format-List -Property Server,InternalClientAuthenticationMethod
105              Get-OutlookAnywhere -server $servername | Set-OutlookAnywhere -InternalClientAuthenticationMethod  Negotiate
106
107              Get-MapiVirtualDirectory -Server $servername | Format-List -Property Server,IISAuthenticationMethods
108              Get-MapiVirtualDirectory -Server $servername | Set-MapiVirtualDirectory -IISAuthenticationMethods oauth,Negotiate
109

PS C:\>             Get-OutlookAnywhere -Server $servername | Format-List -Property Server,InternalClientAuthenticationMethod


Server                            : WS-MX2
InternalClientAuthenticationMethod : Ntlm



PS C:\>             Get-OutlookAnywhere -server $servername | Set-OutlookAnywhere -InternalClientAuthenticationMethod  Negotiate
WARNUNG: Ältere Microsoft Exchange-Versionen als Exchange Server 2013 unterstützen die Clientauthentifizierungsmethode "Verhandeln" nicht. Die Konn
ektivität mit öffentlichen Ordnern und Postfächern, die auf früheren Versionen gehostet werden, kann davon betroffen sein.

PS C:\>             Get-MapiVirtualDirectory -Server $servername | Format-List -Property Server,IISAuthenticationMethods


Server                  : WS-MX2
IISAuthenticationMethods : {Ntlm, OAuth, Negotiate}



PS C:\>             Get-MapiVirtualDirectory -Server $servername | Set-MapiVirtualDirectory -IISAuthenticationMethods oauth,Negotiate

PS C:\>             Get-OutlookAnywhere -Server $servername | Format-List -Property Server,InternalClientAuthenticationMethod


Server                            : WS-MX2
InternalClientAuthenticationMethod : Negotiate
```

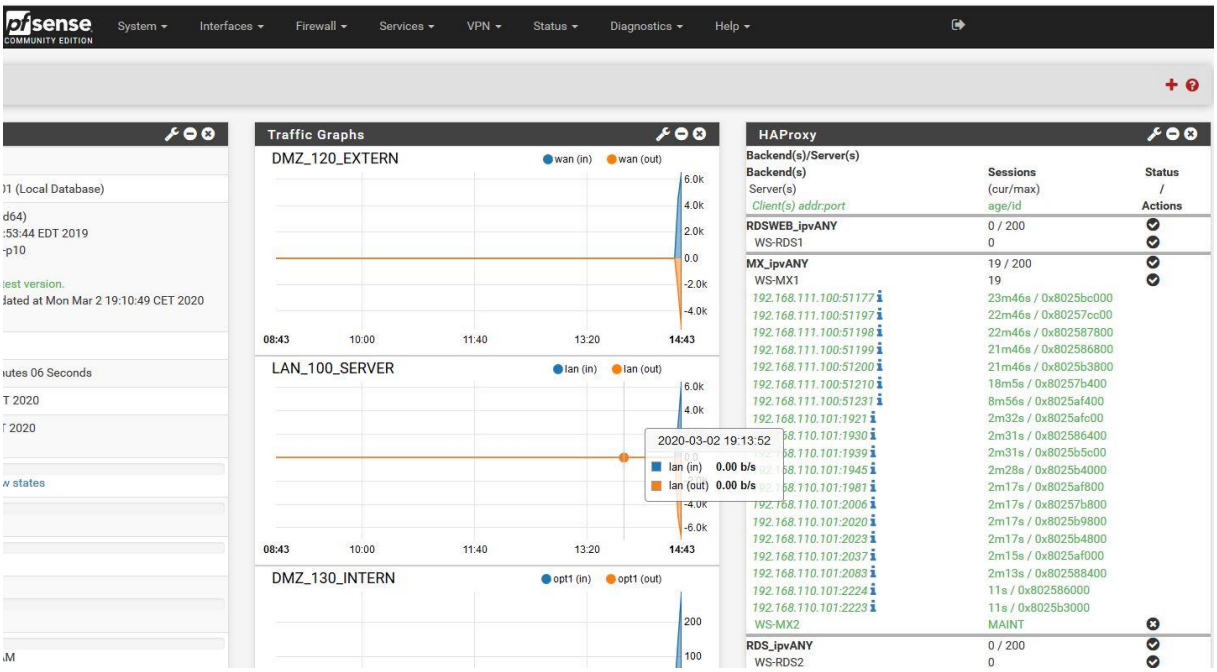
Damit ist der ClientAccessService auf dem neuen WS-MX2 fertig konfiguriert.

Testlauf im Loadbalancer

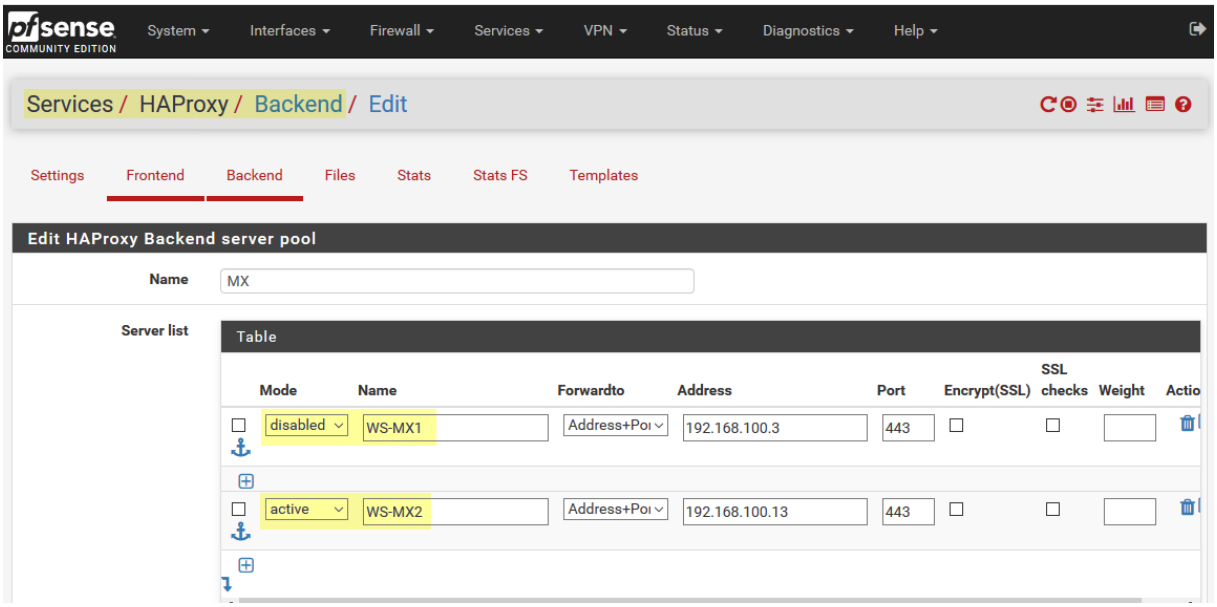
Es wird Zeit für eine Testphase. Dafür starte ich mein Outlook und öffne die Anzeige des Verbindungsstatus. Aktuell bin ich mit dem alten Exchange Server verbunden:



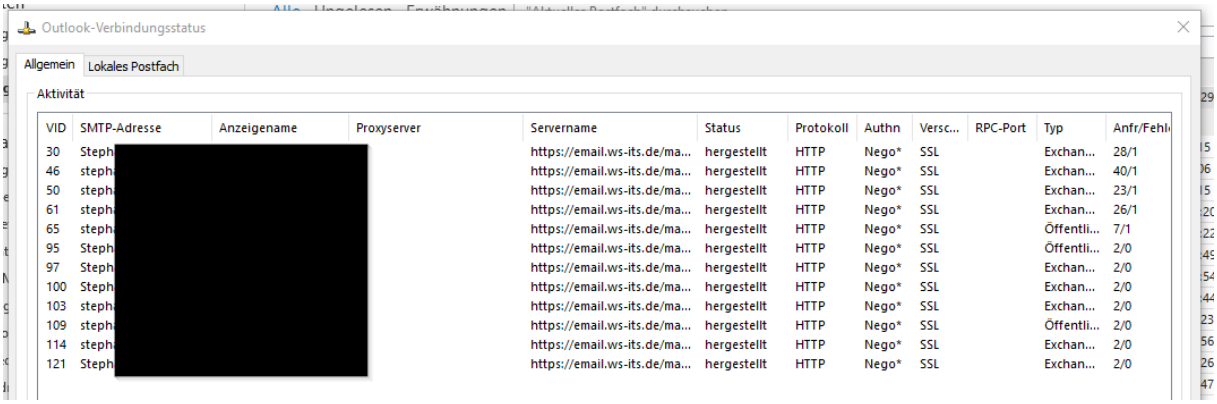
Das war auch zu erwarten, da der Loadbalancer den neuen Server nicht anspricht. Man erkennt bei genauer Betrachtung dessen Maintenance-State:



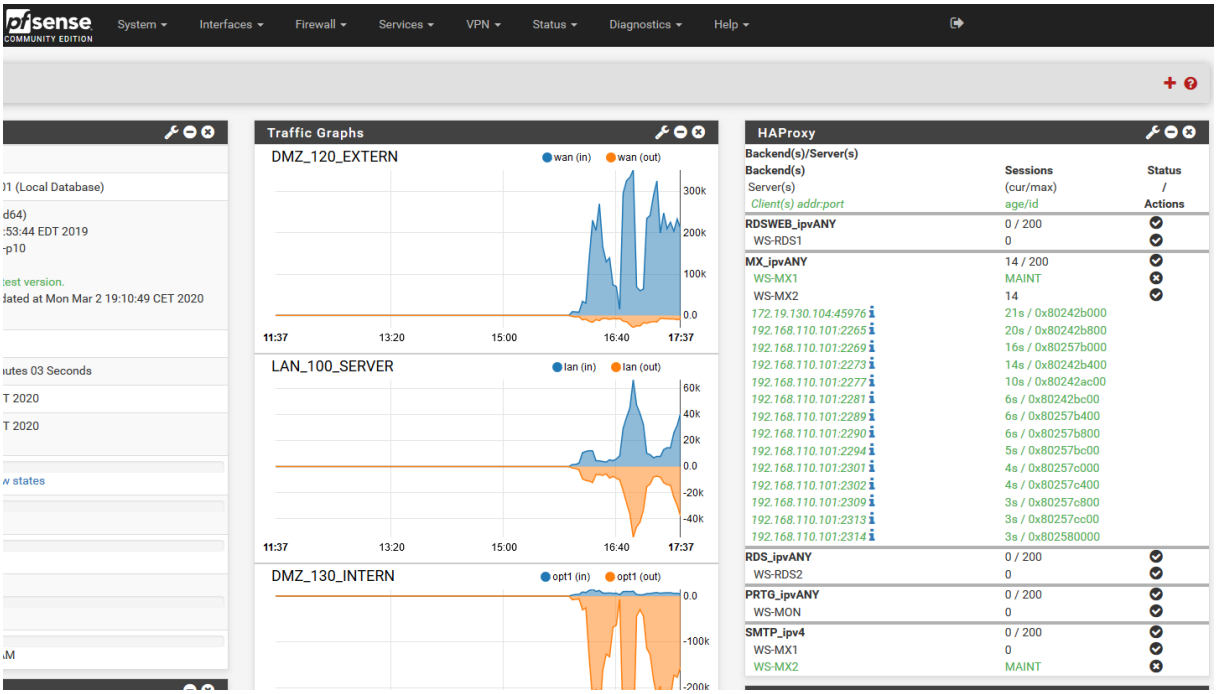
Für meinen Test schwenke ich nun die beiden Server im Loadbalancer: damit verhindere ich den Zugriff auf den alten Mailserver und zwinge die Clients, eine Verbindung zum neuen Server herzustellen. In größeren Umgebungen geht das natürlich etwas anders…



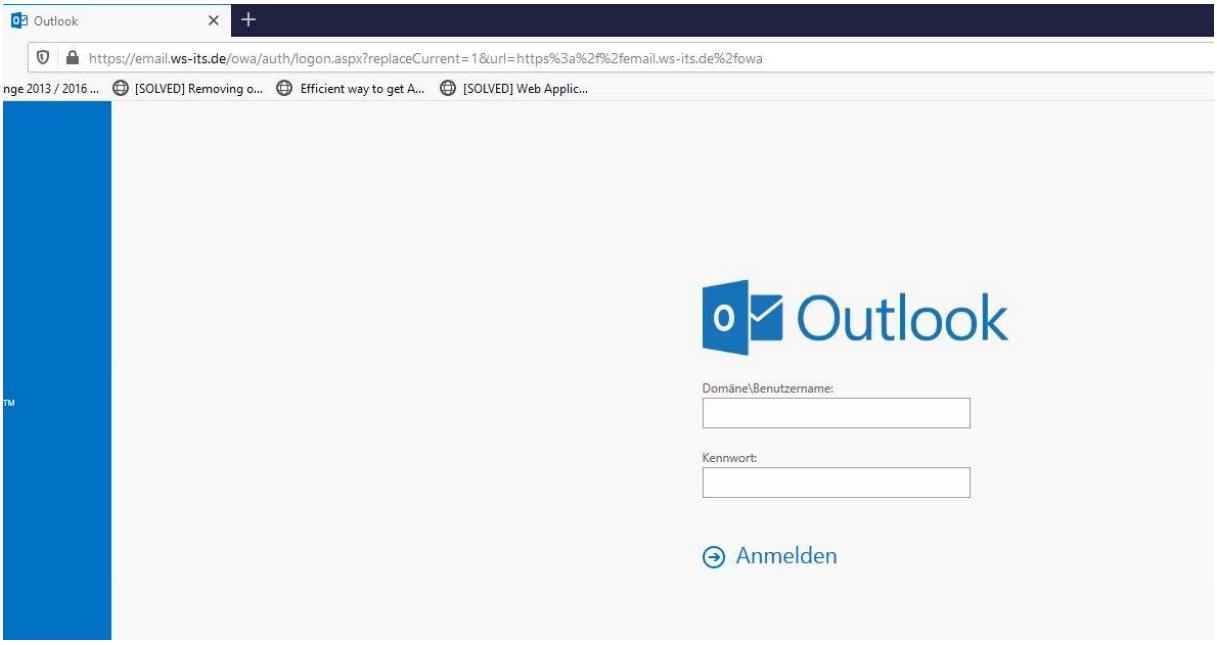
Mein Outlook hat von der Aktion nichts bemerkt. Es ist einfach zu träge und zudem sind die Verbindungen auch nicht dauerhaft etabliert:



Im Loadbalancer sieht man deutlich, dass nun alle Verbindungen den neuen Server verwenden:



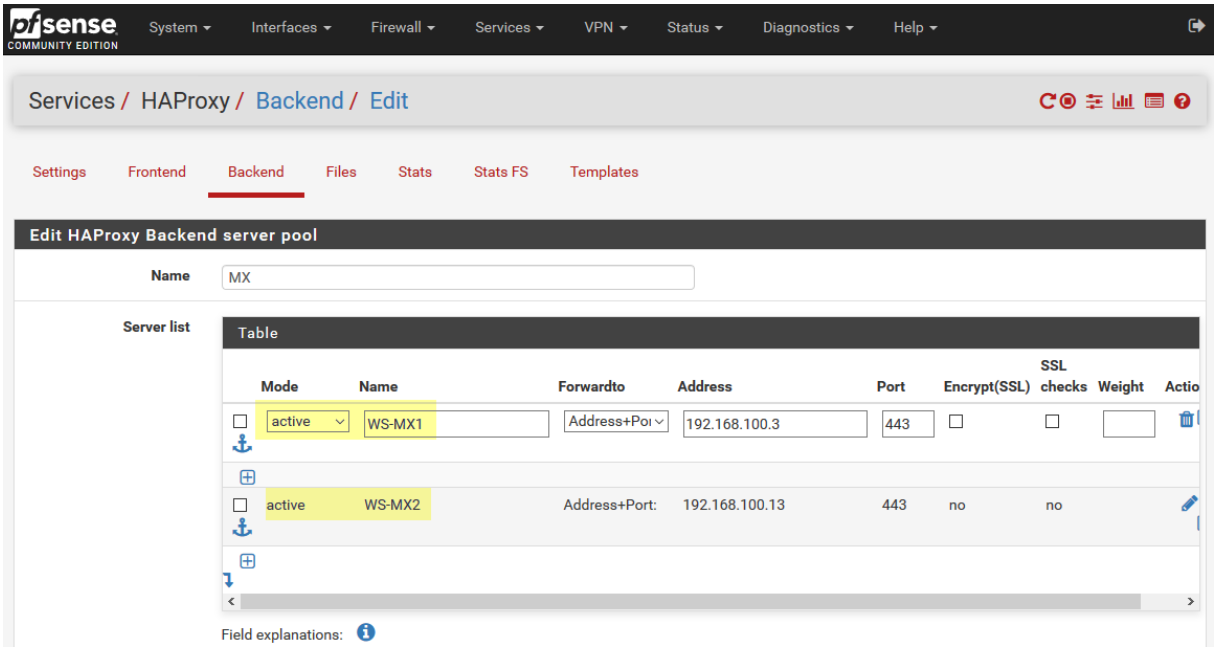
Einen weiteren Test nehme ich mit meinem Browser vor. Hier rufe ich intern die OWA-Webseite auf. Auch diese wird korrekt angezeigt:



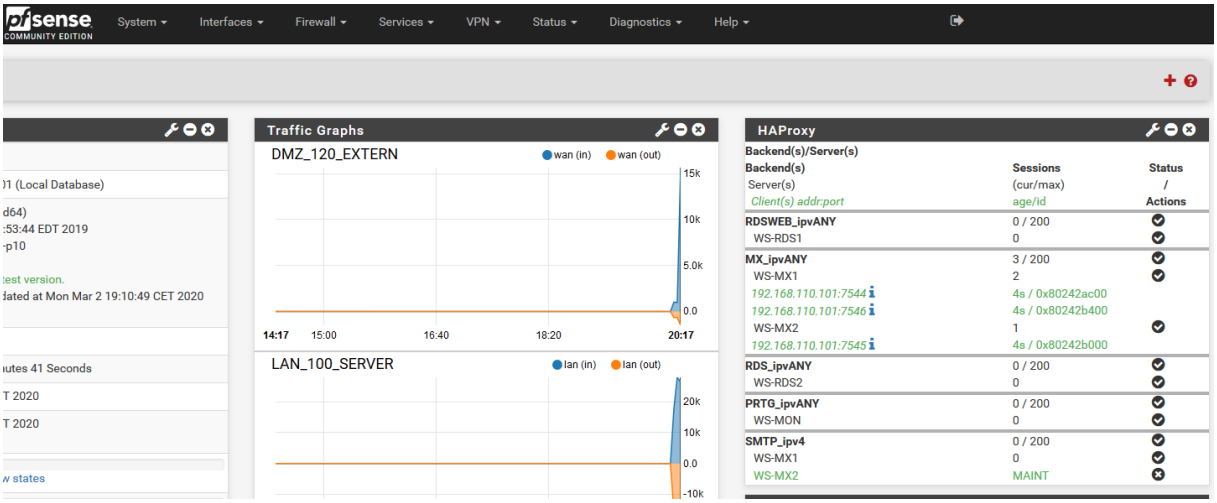
Ebenso bleibt mein Smartphone mit den Mailboxen verbunden. Die Rolle CAS kann also auf dem neuen Server verwendet werden!

Produktivschaltung der CAS-Rolle

So ist der nächste Schritt die Aktivierung beider Mailserver im Loadbalancer:



Jetzt werden die Verbindungen über beide Server verteilt:



Damit ist der Aufbau der ClientAccessServer-Rolle abgeschlossen.

Konfiguration der HTS-Rolle

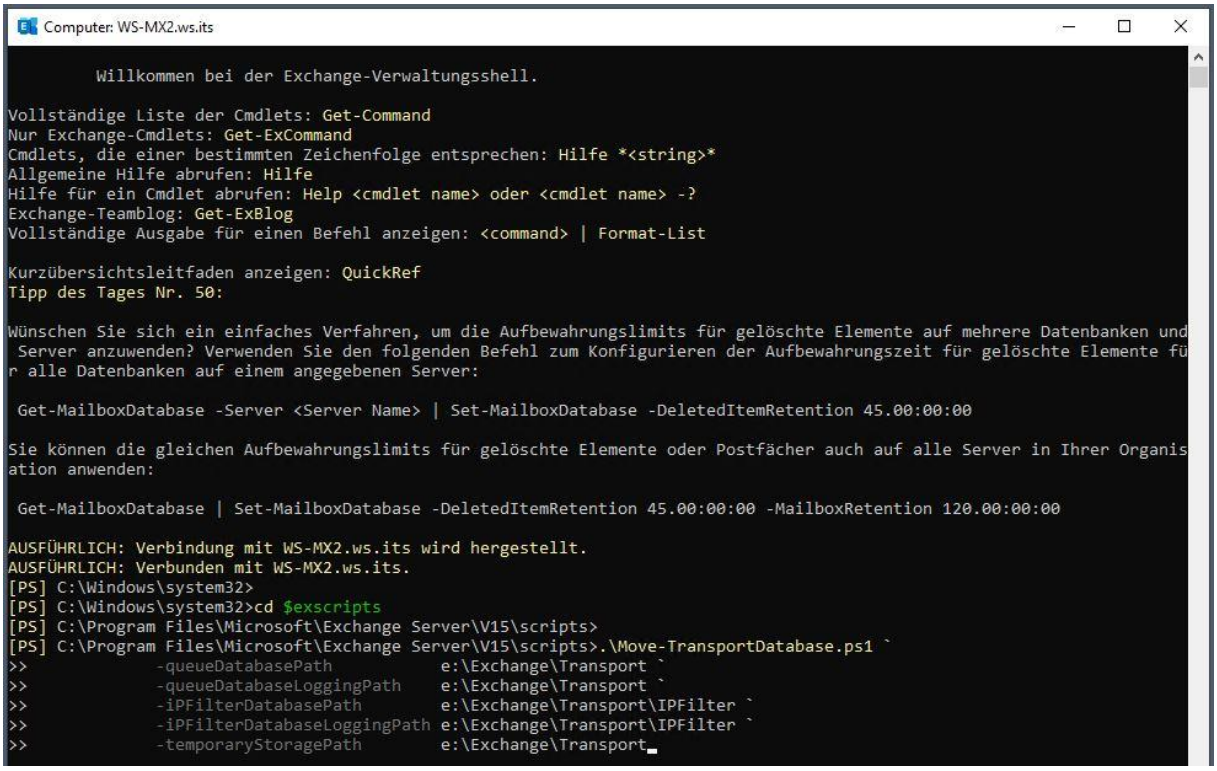
Verschiebung der Transportdatenbank

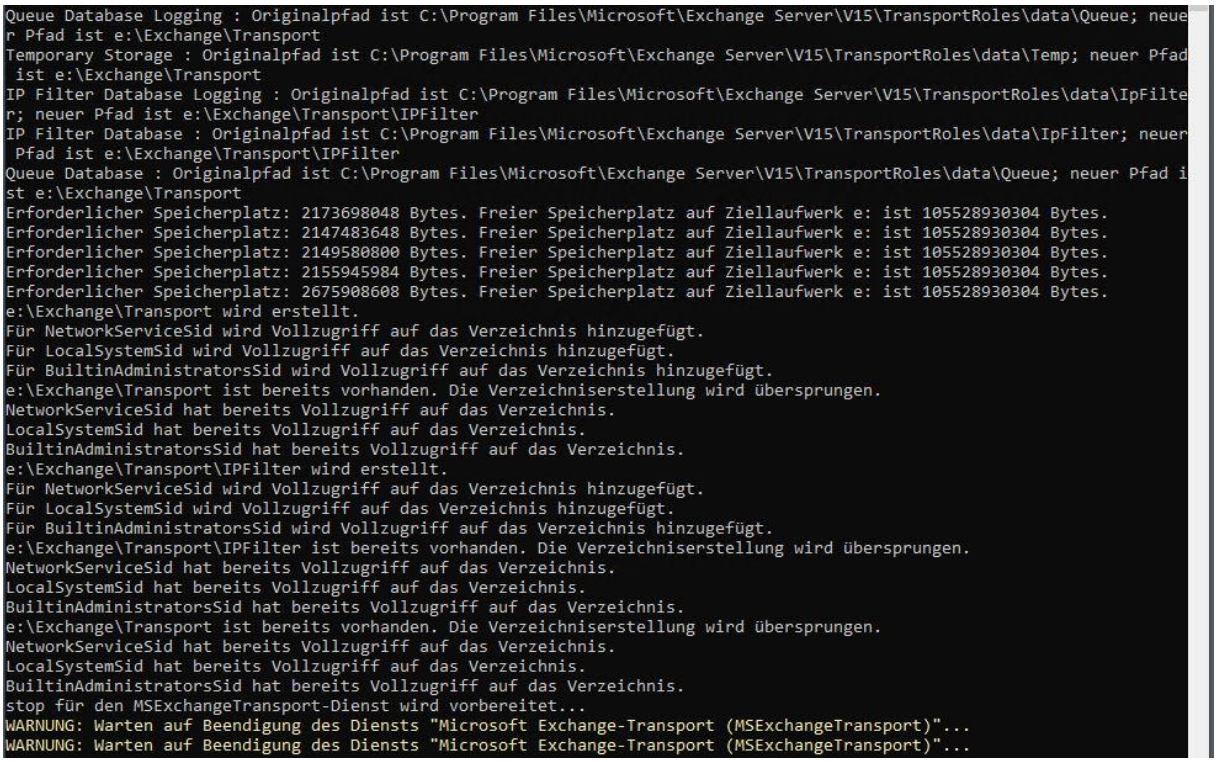
Der neue Mailserver soll auch für den Mailtransport verwendet werden. Diese Rolle übernimmt der im Mailbox-Server integrierte HubTransportService – kurz HTS. Auch diese Rolle konfiguriere ich mit der PowerShell.

Ich beginne mit der Verschiebung der Transport-Datenbank. Diese liegt natürlich im Systemlaufwerk. Sie kann aber durchaus sehr groß werden. Daher verschiebe ich sie auf die Partition mit den anderen Datenbanken. Der dazugehörige PowerShell-Befehl muss aber in der Exchange Management-Shell ausgeführt werden. Mein Script rendert dazu den Befehl:

```
111  # Konfiguration der Rolle HTS
112      # Variablen
113          $servername = "WS-MX2"
114
115      # Verschieben der Transportdatenbank (lokal auf dem Server ausführen)
116          "Set-Location -Path `$exscripts
117          .\Move-TransportDatabase.ps1 `
118              -queueDatabasePath           e:\Exchange\Transport `
119              -queueDatabaseLoggingPath    e:\Exchange\Transport `
120              -iPFilterDatabasePath        e:\Exchange\Transport\IPFilter `
121              -iPFilterDatabaseLoggingPath e:\Exchange\Transport\IPFilter `
122              -temporaryStoragePath        e:\Exchange\Transport"
123
```

Mit Copy & Paste übertrage ich den Befehl in die Management-Shell. Diese muss als Administrator privilegiert ausgeführt werden. Der Transport-Dienst wird dabei kurz angehalten. Das stellt aber kein Problem dar:


Computer: WS-MX2.ws.its
Willkommen bei der Exchange-Verwaltungsshell.
Vollständige Liste der Cmdlets: Get-Command
Nur Exchange-Cmdlets: Get-ExCommand
Cmdlets, die einer bestimmten Zeichenfolge entsprechen: Hilfe *<string>*
Allgemeine Hilfe abrufen: Hilfe
Hilfe für ein Cmdlet abrufen: Help <cmdlet name> oder <cmdlet name> -?
Exchange-Teamblog: Get-ExBlog
Vollständige Ausgabe für einen Befehl anzeigen: <command> | Format-List
Kurzübersichtsleitfaden anzeigen: QuickRef
Tipp des Tages Nr. 50:
Wünschen Sie sich ein einfaches Verfahren, um die Aufbewahrungslimits für gelöschte Elemente auf mehrere Datenbanken und Server anzuwenden? Verwenden Sie den folgenden Befehl zum Konfigurieren der Aufbewahrungszeit für gelöschte Elemente für alle Datenbanken auf einem angegebenen Server:
Get-MailboxDatabase -Server <Server Name> | Set-MailboxDatabase -DeletedItemRetention 45.00:00:00
Sie können die gleichen Aufbewahrungslimits für gelöschte Elemente oder Postfächer auch auf alle Server in Ihrer Organisation anwenden:
Get-MailboxDatabase | Set-MailboxDatabase -DeletedItemRetention 45.00:00:00 -MailboxRetention 120.00:00:00
AUSFÜHRLICH: Verbindung mit WS-MX2.ws.its wird hergestellt.
AUSFÜHRLICH: Verbunden mit WS-MX2.ws.its.
[PS] C:\Windows\system32>
[PS] C:\Windows\system32>cd $exscripts
[PS] C:\Program Files\Microsoft\Exchange Server\V15\scripts>
[PS] C:\Program Files\Microsoft\Exchange Server\V15\scripts>.\Move-TransportDatabase.ps1 `
>> -queueDatabasePath e:\Exchange\Transport `
>> -queueDatabaseLoggingPath e:\Exchange\Transport `
>> -iPFilterDatabasePath e:\Exchange\Transport\IPFilter `
>> -iPFilterDatabaseLoggingPath e:\Exchange\Transport\IPFilter `
>> -temporaryStoragePath e:\Exchange\Transport


Queue Database Logging : Originalpfad ist C:\Program Files\Microsoft\Exchange Server\V15\TransportRoles\data\Queue; neuer Pfad ist e:\Exchange\Transport
Temporary Storage : Originalpfad ist C:\Program Files\Microsoft\Exchange Server\V15\TransportRoles\data\Temp; neuer Pfad ist e:\Exchange\Transport
IP Filter Database Logging : Originalpfad ist C:\Program Files\Microsoft\Exchange Server\V15\TransportRoles\data\IpFilter; neuer Pfad ist e:\Exchange\Transport\IPFilter
IP Filter Database : Originalpfad ist C:\Program Files\Microsoft\Exchange Server\V15\TransportRoles\data\IpFilter; neuer Pfad ist e:\Exchange\Transport\IPFilter
Queue Database : Originalpfad ist C:\Program Files\Microsoft\Exchange Server\V15\TransportRoles\data\Queue; neuer Pfad ist e:\Exchange\Transport
Erforderlicher Speicherplatz: 2173698048 Bytes. Freier Speicherplatz auf Ziellaufwerk e: ist 105528930304 Bytes.
Erforderlicher Speicherplatz: 2147483648 Bytes. Freier Speicherplatz auf Ziellaufwerk e: ist 105528930304 Bytes.
Erforderlicher Speicherplatz: 2149580800 Bytes. Freier Speicherplatz auf Ziellaufwerk e: ist 105528930304 Bytes.
Erforderlicher Speicherplatz: 2155945984 Bytes. Freier Speicherplatz auf Ziellaufwerk e: ist 105528930304 Bytes.
Erforderlicher Speicherplatz: 2675908608 Bytes. Freier Speicherplatz auf Ziellaufwerk e: ist 105528930304 Bytes.
e:\Exchange\Transport wird erstellt.
Für NetworkServiceSid wird Vollzugriff auf das Verzeichnis hinzugefügt.
Für LocalSystemSid wird Vollzugriff auf das Verzeichnis hinzugefügt.
Für BuiltinAdministratorsSid wird Vollzugriff auf das Verzeichnis hinzugefügt.
e:\Exchange\Transport ist bereits vorhanden. Die Verzeichniserstellung wird übersprungen.
NetworkServiceSid hat bereits Vollzugriff auf das Verzeichnis.
LocalSystemSid hat bereits Vollzugriff auf das Verzeichnis.
BuiltinAdministratorsSid hat bereits Vollzugriff auf das Verzeichnis.
e:\Exchange\Transport\IPFilter wird erstellt.
Für NetworkServiceSid wird Vollzugriff auf das Verzeichnis hinzugefügt.
Für LocalSystemSid wird Vollzugriff auf das Verzeichnis hinzugefügt.
Für BuiltinAdministratorsSid wird Vollzugriff auf das Verzeichnis hinzugefügt.
e:\Exchange\Transport\IPFilter ist bereits vorhanden. Die Verzeichniserstellung wird übersprungen.
NetworkServiceSid hat bereits Vollzugriff auf das Verzeichnis.
LocalSystemSid hat bereits Vollzugriff auf das Verzeichnis.
BuiltinAdministratorsSid hat bereits Vollzugriff auf das Verzeichnis.
e:\Exchange\Transport ist bereits vorhanden. Die Verzeichniserstellung wird übersprungen.
NetworkServiceSid hat bereits Vollzugriff auf das Verzeichnis.
LocalSystemSid hat bereits Vollzugriff auf das Verzeichnis.
BuiltinAdministratorsSid hat bereits Vollzugriff auf das Verzeichnis.
stop für den MSExchangeTransport-Dienst wird vorbereitet...
WARNUNG: Warten auf Beendigung des Diensts "Microsoft Exchange-Transport (MSExchangeTransport)"...
WARNUNG: Warten auf Beendigung des Diensts "Microsoft Exchange-Transport (MSExchangeTransport)"...

```
Der MSExchangeTransport-Dienst wurde erfolgreich stopped.
Eine Kopie der ursprünglichen Konfigurationsdatei wird in C:\Program Files\Microsoft\Exchange Server\V15\bin\EdgeTranspo
rt.exe.config.20200305190514.old gespeichert.
Datei trn.log wurde zum Ziel verschoben.
Datei trn00000001.log wurde zum Ziel verschoben.
Datei trntmp.log wurde zum Ziel verschoben.
Datei Trnres00001.jrs wurde zum Ziel verschoben.
Datei Trnres00002.jrs wurde zum Ziel verschoben.
Die Datei Temp.edb wird übersprungen, weil sie nicht vorhanden ist.
Der Queue Database Logging-Pfad wird zu e:\Exchange\Transport aktualisiert.
Der Temporary Storage-Pfad wird zu e:\Exchange\Transport aktualisiert.
Datei trn.log wurde zum Ziel verschoben.
Datei trntmp.log wurde zum Ziel verschoben.
Datei Trnres00001.jrs wurde zum Ziel verschoben.
Datei Trnres00002.jrs wurde zum Ziel verschoben.
Die Datei Temp.edb wird übersprungen, weil sie nicht vorhanden ist.
Der IP Filter Database Logging-Pfad wird zu e:\Exchange\Transport\IPFilter aktualisiert.
Datei IPFiltering.edb wurde zum Ziel verschoben.
Datei trn.chk wurde zum Ziel verschoben.
Der IP Filter Database-Pfad wird zu e:\Exchange\Transport\IPFilter aktualisiert.
Datei mail.que wurde zum Ziel verschoben.
Datei trn.chk wurde zum Ziel verschoben.
Der Queue Database-Pfad wird zu e:\Exchange\Transport aktualisiert.
start für den MSExchangeTransport-Dienst wird vorbereitet...
WARNUNG: Warten auf Start des Diensts "Microsoft Exchange-Transport (MSExchangeTransport)"...
WARNUNG: Warten auf Start des Diensts "Microsoft Exchange-Transport (MSExchangeTransport)"...
Der MSExchangeTransport-Dienst wurde erfolgreich started.
Die Ausführung des Skripts wurde erfolgreich abgeschlossen.
[PS] C:\Program Files\Microsoft\Exchange Server\V15\scripts>
```

Und das war es auch schon.

Aktivierung der AntiSpam und AntiMalware-Features

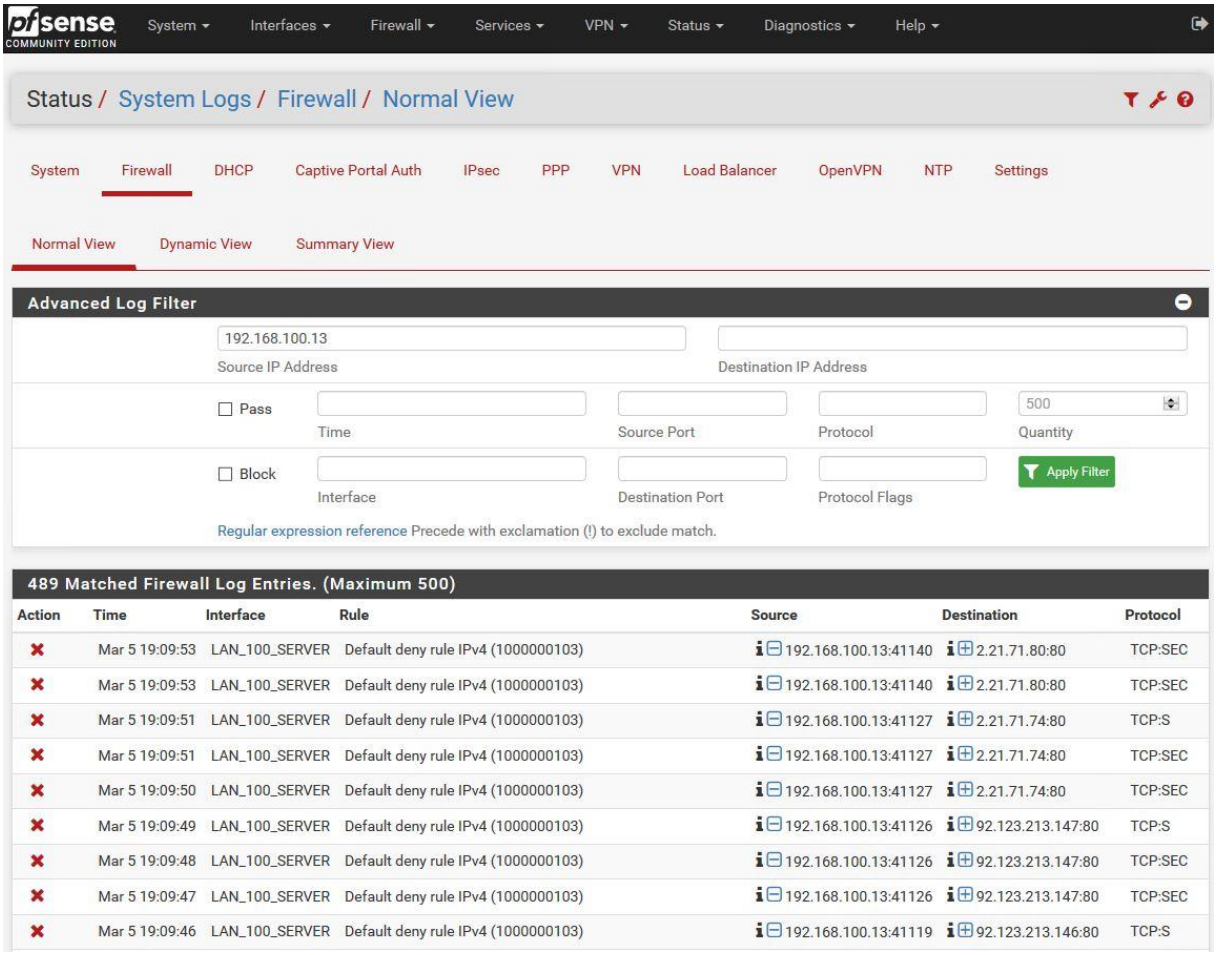
Jetzt kommen die Schutzfunktionen. Über den Mailservice wird viel Spam und ebenso der eine oder andere Schadcode transportiert werden. Beides kann ich nicht gebrauchen. Zur Abwehr verwende ich die eingebauten Features des Exchange Servers. Diese muss ich aber erst mit zwei Exchange-Scripten in der noch offenen Exchange Management-Shell aktivieren:

```
124     # Aktivierung der TransportAgents für Antimalware und Antispam (lokal auf dem Server ausführen)
125         "Set-Location -Path `$exscripts
126         .\Enable-AntimalwareScanning.ps1
127         .\Install-AntiSpamAgents.ps1
128
129         Restart-Service MSExchangeTransport
130         Restart-Service MSExchangeFrontEndTransport
131         "
132
133         Get-TransportAgent -Identity $servername
134
```

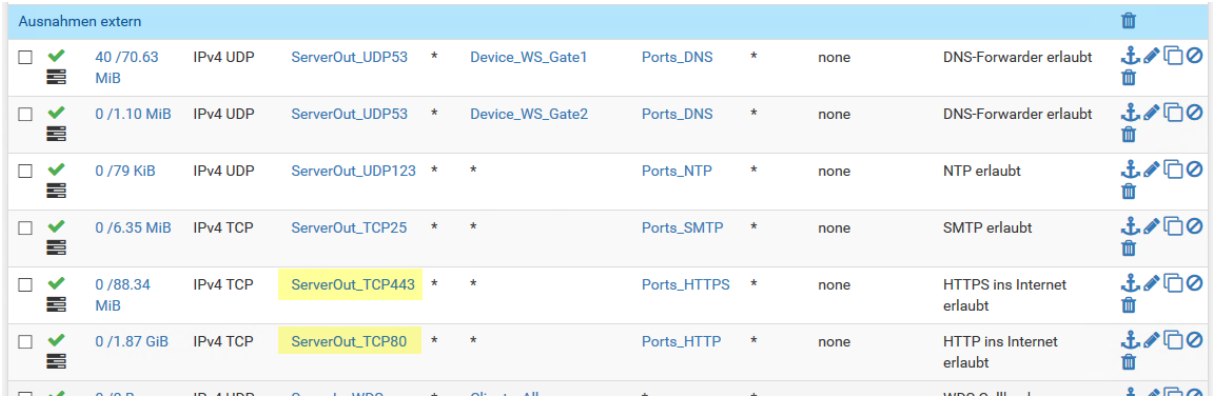
Das erste Script möchte eine Verbindung zum Internet aufbauen, um neue Module herunterzuladen. Das scheint aber nicht zu funktionieren:

```
Computer: WS-MX2.ws.its
[PS] C:\Program Files\Microsoft\Exchange Server\V15\scripts>.\Enable-AntimalwareScanning.ps1
Antischadsoftware-Module werden aktualisiert. Dies kann einige Minuten dauern.
Es wird auf Module geprüft, die nach 27.02.2020 19:07:21 aktualisiert wurden.
"Microsoft" wird aktualisiert. Letztes Update: 01.01.1900 01:00:00
...
Es wird auf Module geprüft, die nach 27.02.2020 19:07:21 aktualisiert wurden.
"Microsoft" wird aktualisiert. Letztes Update: 01.01.1900 01:00:00
...
Es wird auf Module geprüft, die nach 27.02.2020 19:07:21 aktualisiert wurden.
"Microsoft" wird aktualisiert. Letztes Update: 01.01.1900 01:00:00
...
Es wird auf Module geprüft, die nach 27.02.2020 19:07:21 aktualisiert wurden.
"Microsoft" wird aktualisiert. Letztes Update: 01.01.1900 01:00:00
...
Es wird auf Module geprüft, die nach 27.02.2020 19:07:21 aktualisiert wurden.
"Microsoft" wird aktualisiert. Letztes Update: 01.01.1900 01:00:00
```

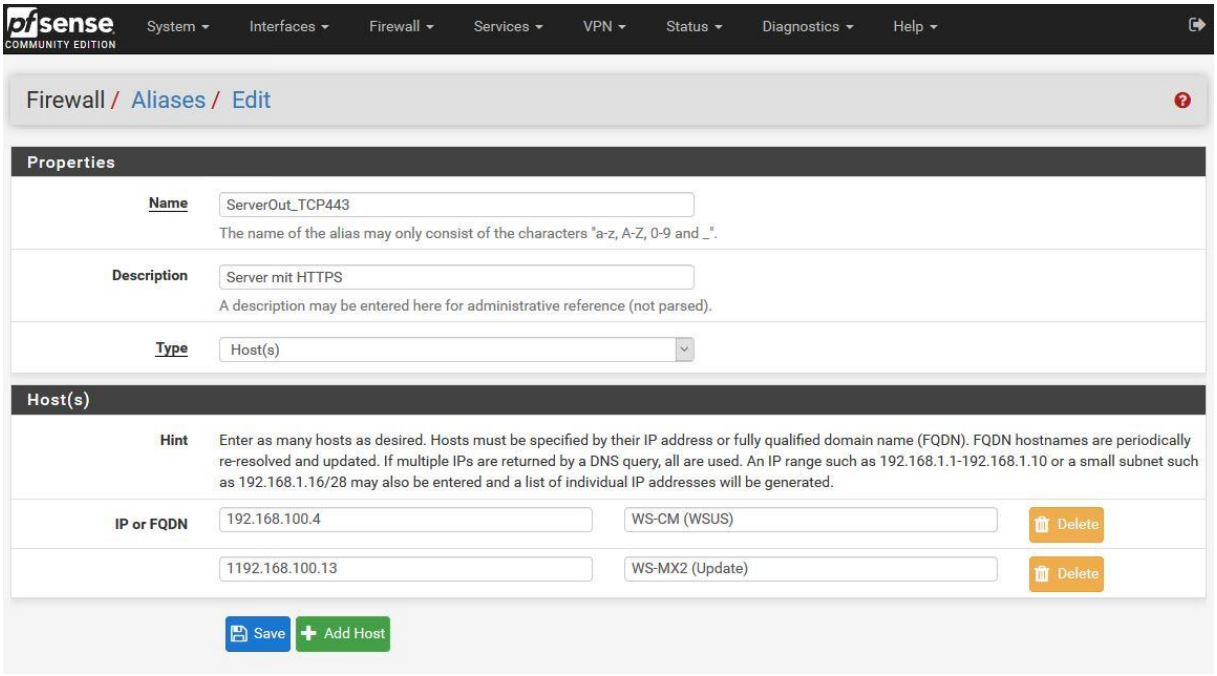
Die Ursache ist schnell gefunden: Meine Firewall blockiert die Downloads, da meine Mailserver wie alle anderen auch per Default keinen Internetzugriff haben:



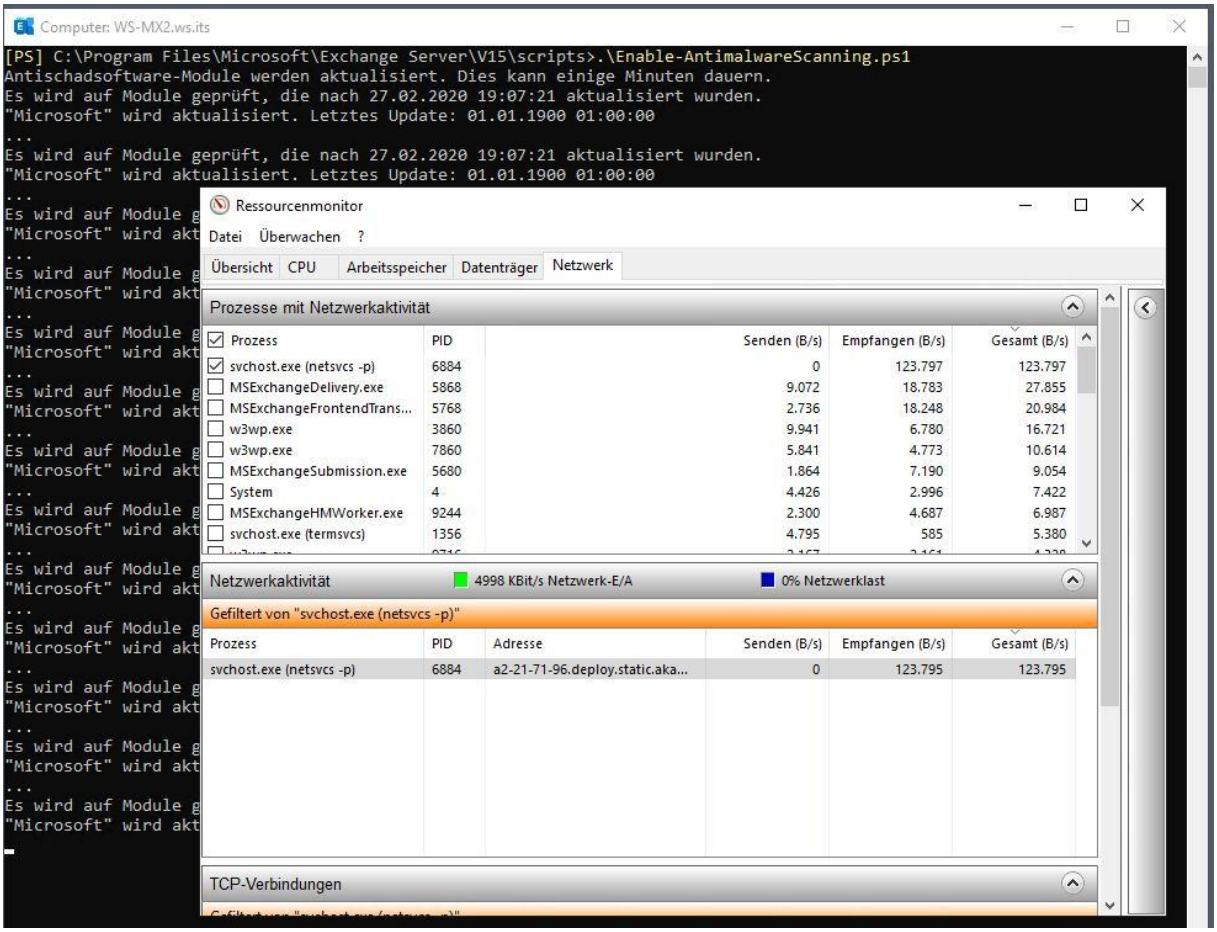
Fürs Erste erlaube ich die Protokolle http und https. Später werde ich die erforderlichen URL zusätzlich beschränken. Diese beiden Regeln in meiner PFSense-Firewall sind für den Internet-Zugriff aus dem Server-VLAN zuständig. Die Quellserver habe ich dabei als Alias angelegt:



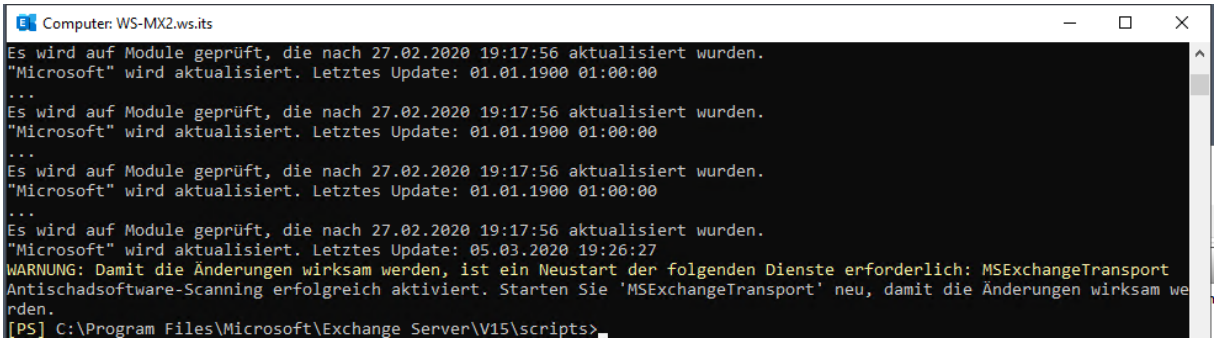
Nun editiere ich beide Aliase und trage den neuen Mailserver mit seiner IP-Adresse ein:



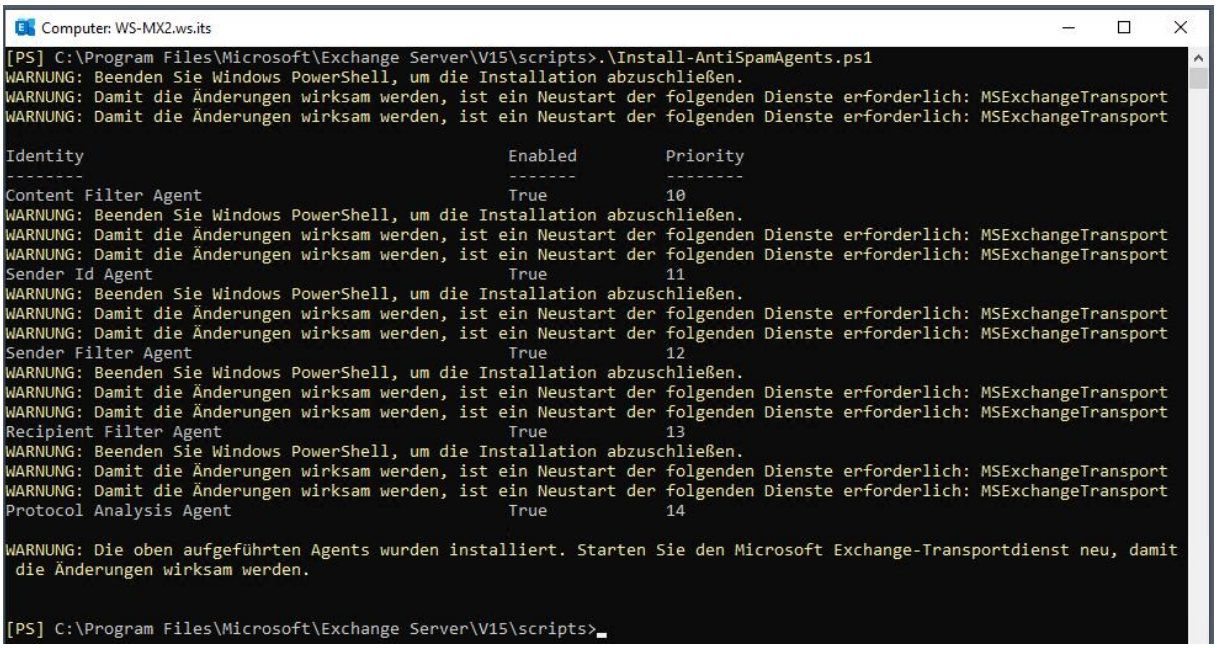
Ein Apply später wird die erste Verbindung aufgebaut:



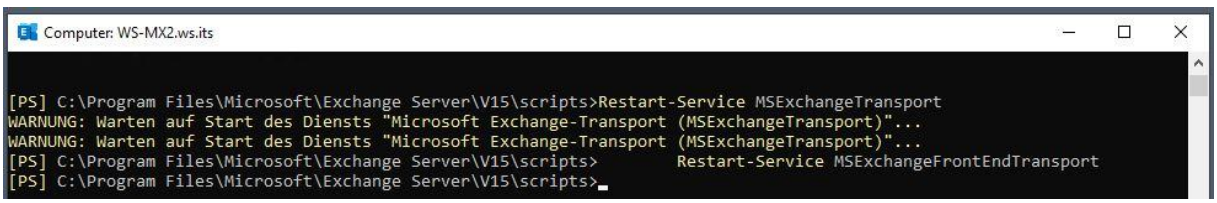
Es dauert aber noch ein paar Minuten, bis das Script alles abgeschlossen hat:



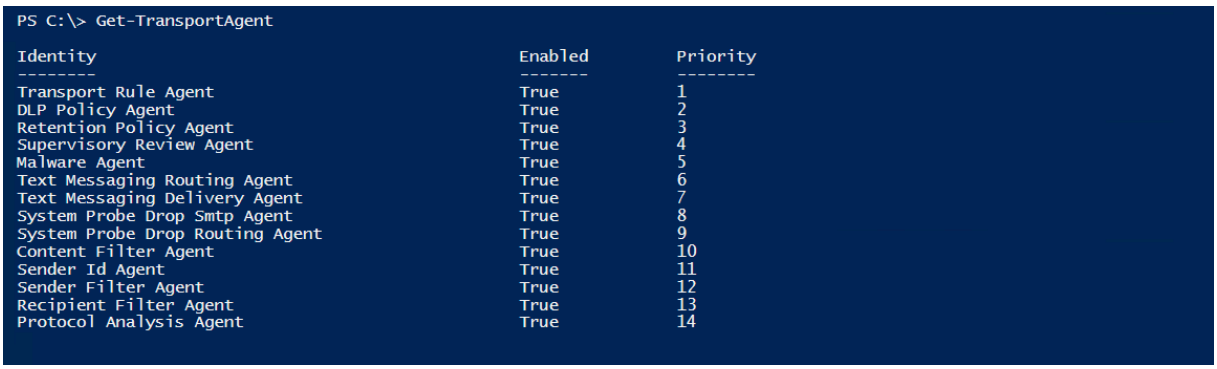
Jetzt führe ich das Script zur Installation des AntiSpam-Agents aus:



Beide Scripte erfordern einen Neustart der Transport-Dienste:



Danach sind die Schutzkomponenten im HTS integriert:



Die Konfiguration ist bereits im Active Directory abgelegt. Daher muss hier nichts angepasst werden.

Konfiguration der Konnektoren

Nun sind die Konnektoren für den Mailfluss an der Reihe. Ich editiere zuerst den „Default Frontend“-Konnektor. Dieser nimmt normalerweise von allen IP-Adressen auf Port 25 Mails entgegen. Dabei wird aber immer der FQDN des Mailservers übertragen. Mein Serverzertifikat ist aber nur für den externen Namen email.ws-its.de ausgestellt. Den FQDN des System-Konnektors kann man nicht verändern. Daher reduziere ich im ersten Befehlsblock die IP-Adressen des Konnektors auf interne Adressen.

Im zweiten Block aktiviere ich die Protokollierung auf allen Default-Konnektoren. Damit kann ich später Probleme im Mailfluss leichter erkennen.

Im dritten Block baue ich einen neuen Empfangs-Konnektor. Dieser darf von allen IPv4-Adressen verwendet werden. Und als FQDN soll mein Mailserver den richtigen Namen – der auch im TLS-Zertifikat steht – verwenden.

Sowohl mein Monitoring-Server mit der IP-Adresse 192.168.100.18 als auch der Loadbalancer mit der IP-Adresse 192.168.100.250 senden regelmäßige Verbindungsaufbauten zum Mailserver. Diese will ich nicht in meinem Logging haben. Daher baue ich für beide IPs einen weiteren Empfangs-Konnektor. Dieser hat dann keine Protokollierung.

```
135     # Konfiguration der Empfangskonnektoren
136         Get-ReceiveConnector -Server $servername |
137             Where-Object { $_.identity -like '*Default Frontend*' } |
138                 Set-ReceiveConnector -RemoteIPRanges '192.168.100.0/24','192.168.101.0/24','192.168.111.0/24'
139
140         Get-ReceiveConnector -Server $servername |
141             Where-Object { $_.identity -like '*Default *' } |
142                 Set-ReceiveConnector -ProtocolLoggingLevel 'verbose'
143
144         New-ReceiveConnector `
145                 -Name                   'Mails-vom-Internet' `
146                 -MaxMessageSize         50MB `
147                 -Enabled                $true `
148                 -ProtocolLoggingLevel   'verbose' `
149                 -AuthMechanism          'Tls' `
150                 -Fqdn                   'email.ws-its.de' `
151                 -PermissionGroups       'AnonymousUsers' `
152                 -RemoteIPRanges         '0.0.0.0-255.255.255.255' `
153                 -Bindings               '0.0.0.0:25' `
154                 -Server                 $servername `
155                 -TransportRole          'FrontEndTransport'
156
157         New-ReceiveConnector `
158                 -Name                   'ProbeMails' `
159                 -Enabled                $true `
160                 -ProtocolLoggingLevel   'none' `
161                 -AuthMechanism          'Tls' `
162                 -PermissionGroups       'AnonymousUsers' `
163                 -RemoteIPRanges         '192.168.100.18','192.168.100.250' `
164                 -Bindings               '0.0.0.0:25' `
165                 -Server                 $servername `
166                 -TransportRole          'FrontEndTransport' `
167                 -Comment 'Probemails ohne Logging'
168
169         Get-ReceiveConnector | Format-Table -Property Identity,Bindings,Enabled,ProtocolLoggingLevel
170
```

So schicke ich die Befehle ab. Und hier sieht man die Ergebnisse:

```
PS C:\>         Get-ReceiveConnector -Server $servername |
            Where-Object { $_.identity -like '*Default Frontend*' } |
                Set-ReceiveConnector -RemoteIPRanges '192.168.100.0/24','192.168.101.0/24','192.168.111.0/24'


PS C:\>         Get-ReceiveConnector -Server $servername |
            Where-Object { $_.identity -like '*Default *' } |
                Set-ReceiveConnector -ProtocolLoggingLevel 'verbose'

WARNUNG: Der Befehl wurde erfolgreich abgeschlossen, es wurden jedoch keine Einstellungen von 'WS-MX2\Default Frontend
WS-MX2' geändert.

PS C:\>         New-ReceiveConnector `
                -Name                'Mails-vom-Internet' `
                -MaxMessageSize      50MB `
                -Enabled             $true `
                -ProtocolLoggingLevel 'verbose' `
                -AuthMechanism       'Tls' `
                -Fqdn                'email.ws-its.de' `
                -PermissionGroups    'AnonymousUsers' `
                -RemoteIPRanges      '0.0.0.0-255.255.255.255' `
                -Bindings            '0.0.0.0:25' `
                -Server              $servername `
                -TransportRole       'FrontEndTransport'


Identity                 Bindings     Enabled
--------                 --------     -------
WS-MX2\Mails-vom-Internet {0.0.0.0:25} True


PS C:\>         New-ReceiveConnector `
                -Name                'ProbeMails' `
                -Enabled             $true `
                -ProtocolLoggingLevel 'none' `
                -AuthMechanism       'Tls' `
                -PermissionGroups    'AnonymousUsers' `
                -RemoteIPRanges      '192.168.100.18','192.168.100.250' `
                -Bindings            '0.0.0.0:25' `
                -Server              $servername `
                -TransportRole       'FrontEndTransport' `
                -Comment 'Probemails ohne Logging'


Identity          Bindings     Enabled
--------          --------     -------
WS-MX2\ProbeMails {0.0.0.0:25} True


PS C:\>
```

Es fehlt noch der Zugriff auf den Sende-Konnektor. Dieser ist serverübergreifend. Also muss ich nur den neuen Server in die Liste der Source-Transport-Server mit aufnehmen:

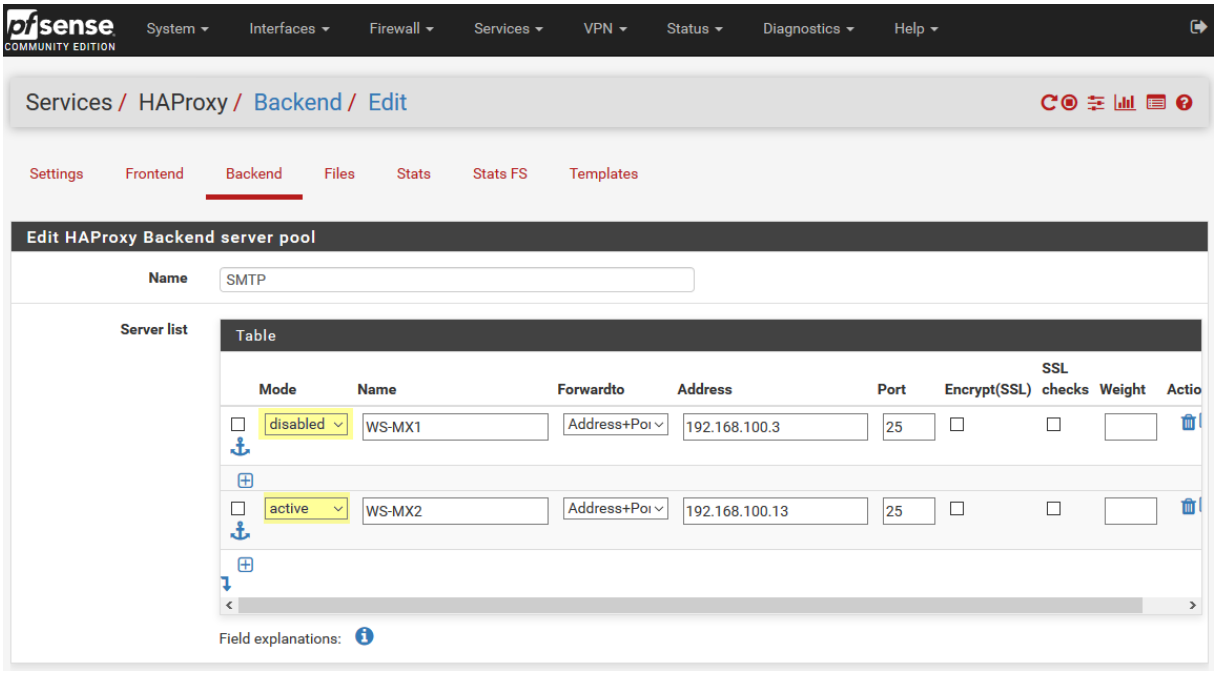
```
171     # Konfiguration des Sendekonnektors
172         Get-SendConnector | Set-SendConnector -SourceTransportServers 'ws-mx1','ws-mx2'
173
```

```
PS C:\>         Get-SendConnector | Set-SendConnector -SourceTransportServers 'ws-mx1','ws-mx2'

PS C:\>
```

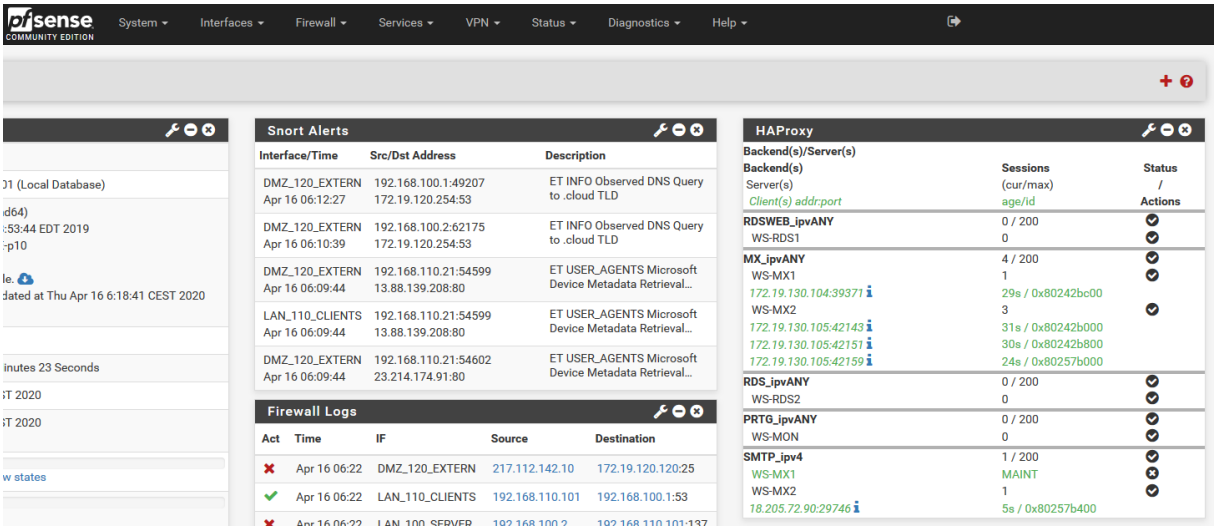
Damit kann mein neuer WS-MX2 Mails senden und empfangen.

Testlauf und Produktivschaltung

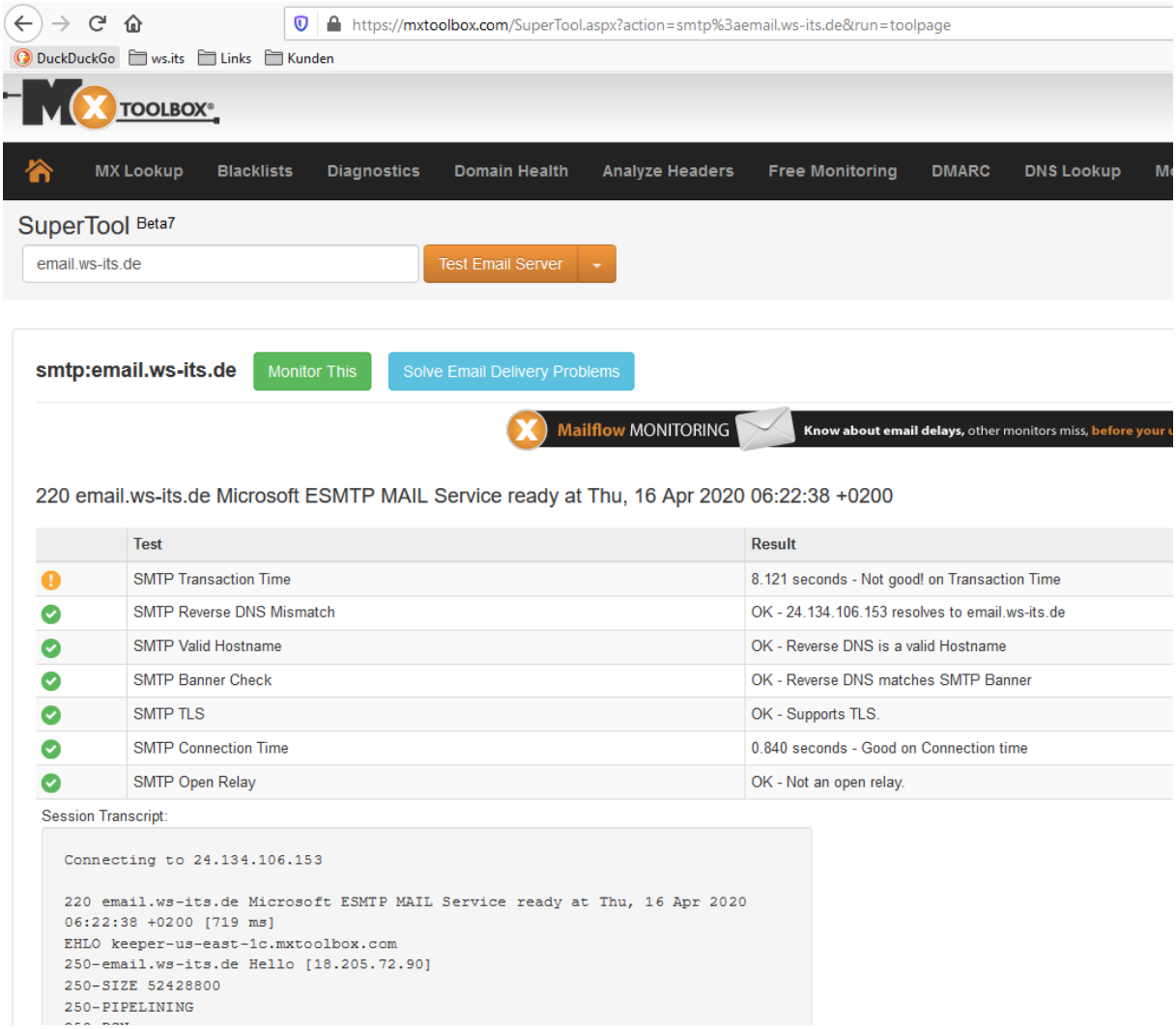
Aber auch hier soll ein Testlauf eventuelle Konfigurationsfehler aufdecken. Mails aus dem Internet werden ebenfalls durch den Loadbalancer geschickt. In diesem ist der neue Server momentan im Wartungszustand. Wie beim CAS verdrehe ich auch hier die Konfigurationen. Weitere Mails kommen jetzt ausschließlich am neuen Exchange Server an:



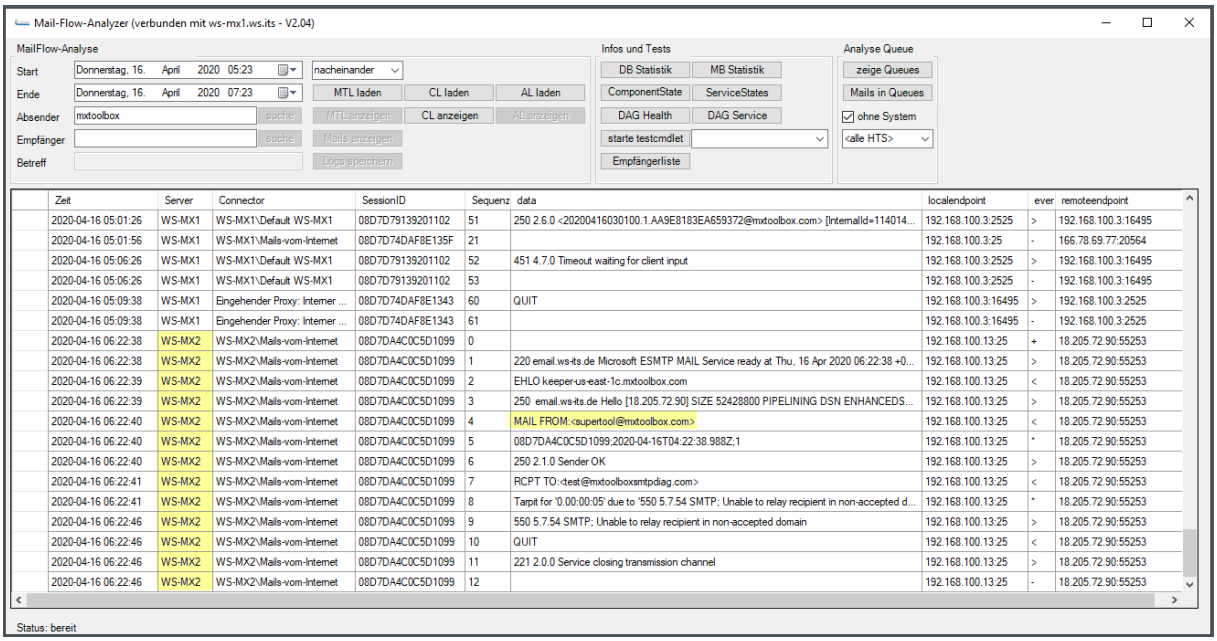
Das ist sehr schön im unteren, rechten Bereich erkennbar. Der alte Server hat Pause und der neue Server unterhält sich mit einem Server im Internet:



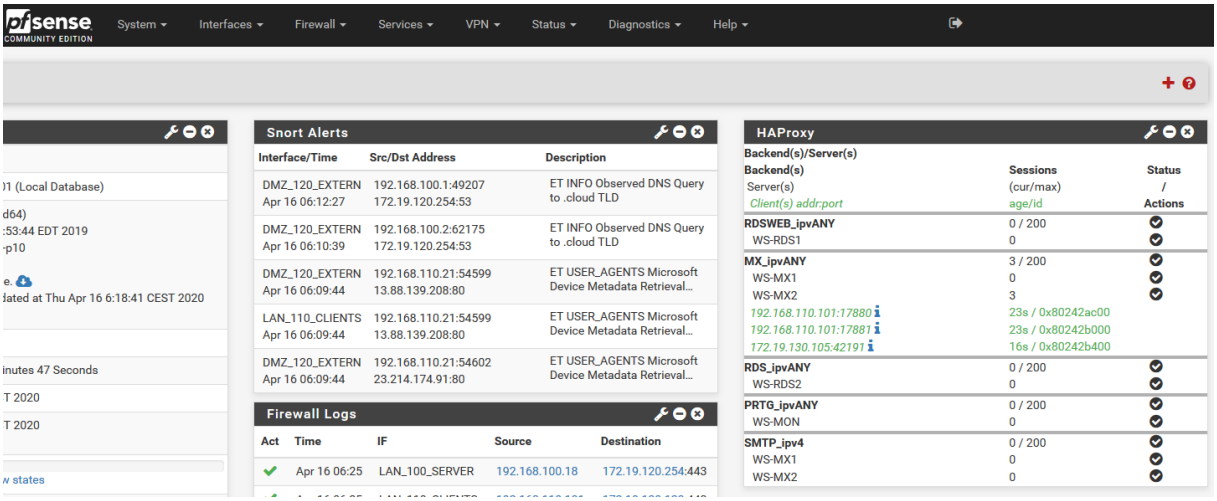
Das ist kein Zufall. Zuvor habe ich mit dem Test-Tool von mxtoolbox einen Mailversand vorbereitet. In der Webausgabe kann ich mir das Ergebnis ansehen:



Und auch mit meiner PowerShell-GUI kann ich den Mailfluss grafisch beobachten:



Hier funktioniert alles. Daher schalte ich die Rolle frei und aktiviere im Loadbalancer wieder beide Server:

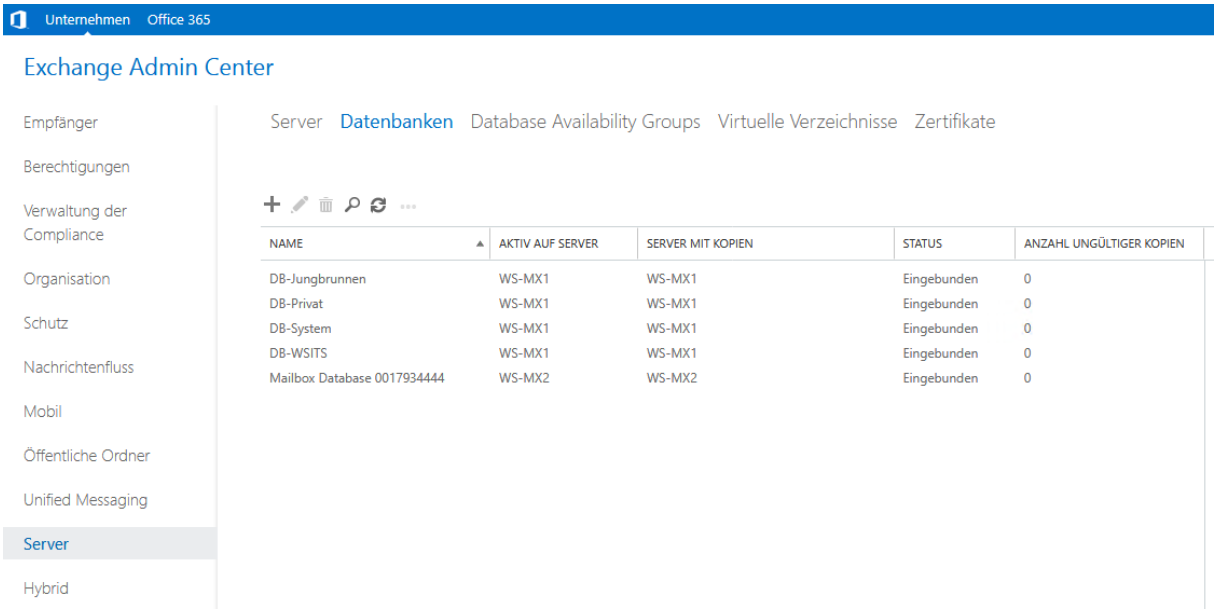


Damit ist auch der Mailfluss konfiguriert.

Konfiguration der MBS-Rolle

Konfiguration der neuen Mailbox-Datenbanken

Es folgt die dritte Komponente eines Exchange Servers: die Mailbox-Server-Rolle – oft auch als MBS abgekürzt. Auf dem alten Server existieren aktuell 4 Postfachdatenbanken. Aber auch der neue Server hat eine Default-Database mitgebracht:



Ich muss die Mailboxen, die in den alten Datenbanken enthalten sind, auf den neuen Server verschieben. Eine gemeinsame Datenbankverfügbarkeitsgruppe scheidet wegen der unterschiedlichen Betriebssysteme (Win 2016 und Win2019) und wegen der unterschiedlichen Exchange Server Versionen aus. Also erstelle ich neue Datenbanken auf dem neuen Server und verschiebe die Mailboxen. Danach kann ich die alten Datenbanken entfernen.

Wichtig ist aber, dass jede Datenbank einen eindeutigen Bezeichner hat. Da ich meine alten Namen später wiederverwenden möchte, hänge ich an jede neue DB einfach ein „-neu“ an. Die Pfade der Dateien im Dateisystem kann ich aber schon an den Zielnamen anpassen.

Die neue Default-Datenbank benötige ich nicht. Daher möchte ich sie löschen:

```
# Konfiguration der Rolle MBS
    # Entfernen der Standard-Datenbanke
        Get-MailboxDatabase -Server WS-MX2 | Remove-MailboxDatabase -Verbose
```

```
PS C:\>         Get-MailboxDatabase -Server WS-MX2 | Remove-MailboxDatabase -Verbose
Diese Postfachdatenbank enthält mindestens ein Postfach, einen Postfachplan, ein Archivpostfach, ein Postfach für öffentliche
Ordner oder ein Vermittlungspostfach oder ein Überwachungspostfach. Führen Sie den Befehl "Get-Mailbox -Database <Database
ID>" aus, um eine Liste aller Postfächer in der Datenbank abzurufen. Führen Sie den Befehl "Get-MailboxPlan" aus, um eine
Liste aller Postfachpläne in dieser Datenbank abzurufen. Führen Sie den Befehl "Get-Mailbox -Database <Database ID> -Archive"
aus, um eine Liste mit Archivpostfächern in dieser Datenbank abzurufen. Um eine Liste aller Postfächer für öffentliche Ordner
in dieser Datenbank abzurufen, führen Sie den Befehl "Get-Mailbox -Database <Database ID> -PublicFolder" aus. Führen Sie den
Befehl "Get-Mailbox -Database <Database ID> -Arbitration" aus, um eine Liste aller Vermittlungspostfächer in dieser Datenbank
abzurufen. Um eine Liste aller Überwachungspostfächer in dieser Datenbank abzurufen, führen Sie den Befehl "Get-Mailbox
-Database <Database ID> -AuditLog" aus. Wenn Sie ein Postfach, bei dem es sich nicht um ein Vermittlungspostfach handelt,
deaktivieren möchten, um die Postfachdatenbank löschen zu können, führen Sie den Befehl "Disable-Mailbox <Mailbox ID>" aus.
Zum Deaktivieren eines Archivpostfachs, um die Postfachdatenbank löschen zu können, führen Sie den Befehl "Disable-Mailbox
<Mailbox ID> -Archive" aus. Zum Deaktivieren eines Postfachs für öffentliche Ordner, um die Postfachdatenbank zu löschen,
führen Sie den Befehl "Disable-Mailbox <Mailbox ID> -PublicFolder" aus. Um ein Überwachungspostfach zu deaktivieren, damit Sie
die Postfachdatenbank löschen können, führen Sie den Befehl "Get-Mailbox -AuditLog | Disable-Mailbox" aus.
Vermittlungspostfächer sollten auf einen anderen Server verschoben werden. Führen Sie hierzu den Befehl "New-MoveRequest
<parameters>" aus. Falls es sich um den letzten Server in der Organisation handelt, führen Sie den Befehl "Disable-Mailbox
<Mailbox ID> -Arbitration -DisableLastArbitrationMailboxAllowed" aus, um das Vermittlungspostfach zu deaktivieren.
Postfachpläne sollten auf einen anderen Server verschoben werden. Führen Sie hierzu den Befehl "Set-MailboxPlan <MailboxPlan
ID> -Database <Database ID>" aus.
    + CategoryInfo          : InvalidOperation: (Mailbox Database 0017934444:DatabaseIdParameter) [Remove-MailboxDatabase], As
   sociatedUserMailboxExistException
    + FullyQualifiedErrorId : [Server=WS-MX2,RequestId=0c77ee5e-49dc-435d-9c57-6d8af68b19d7,TimeStamp=16.04.2020 04:32:48] [Fa
   ilureCategory=Cmdlet-AssociatedUserMailboxExistException] 3A662F0,Microsoft.Exchange.Management.SystemConfigurationTasks.R
  emoveMailboxDatabase
    + PSComputerName        : ws-mx2.ws.its
```

Da ist aber schon etwas enthalten. Mit der PowerShell lässt sich das Geheimnis schnell lüften:

```
            Get-MailboxStatistics -Database (Get-MailboxDatabase -Server WS-MX2).name
```

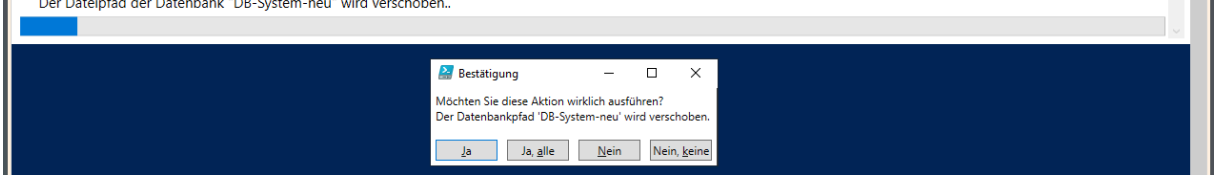
```
PS C:\>             Get-MailboxStatistics -Database (Get-MailboxDatabase -Server WS-MX2).name

DisplayName               ItemCount    StorageLimitStatus                                                LastLogonTime
-----------               ---------    ------------------                                                -------------
SystemMailbox{6b95ce58-cc 16
fe-436c-b952-5d1f77a4f73a
}
HealthMailbox-WS-MX2-Mail 219                                                                            16.04.2020 01:01:12
box-Database-0017934444
In-Situ-Archiv -HealthMai 36
lbox-WS-MX2-Mailbox-Datab
ase-0017934444
Microsoft Exchange        42
```

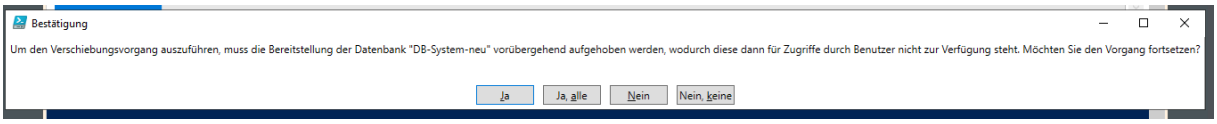
OK, dann plane ich um. Die Datenbank erhält einen neuen Namen „DB-System-neu“ und wird auf die Partition der Exchange Datenbanken verschoben. So ist die erste der 4 neuen DBs fertig. Hier gibt es den neuen Namen:

```
    # Umbenennen und Verschieben der Standard-Datenbank
        Get-MailboxDatabase -Server "WS-MX2" | Set-MailboxDatabase -Name "DB-System-neu"
```

Und hier ein neuen Zuhause:

```
        Move-DatabasePath `
            -Identity       "DB-System-neu" `
            -EdbFilePath    "E:\Exchange\DB-System\DB-System.edb" `
            -LogFolderPath "E:\Exchange\DB-System\Logs"
```



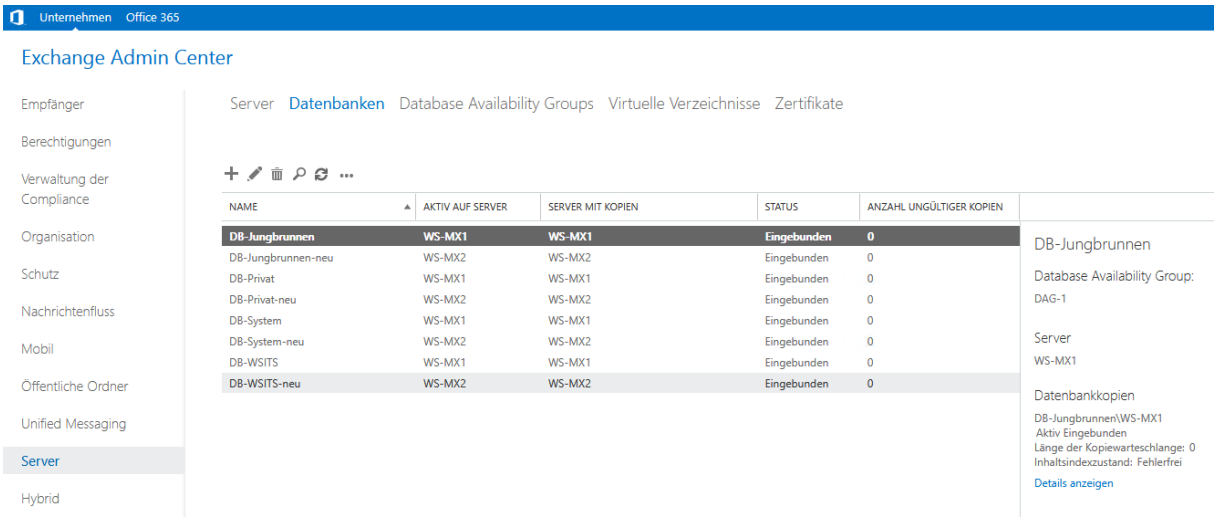


Jetzt benötige ich noch 3 weitere Datenbanken. Nebenbei bemerkt: an einer solchen Stelle ist ein Redesign der Mailboxdatenbanken sehr gut platzierbar. Mein Layout passt mir aber. Die 3 neuen Datenbanken sind mit wenigen Zeilen angelegt:

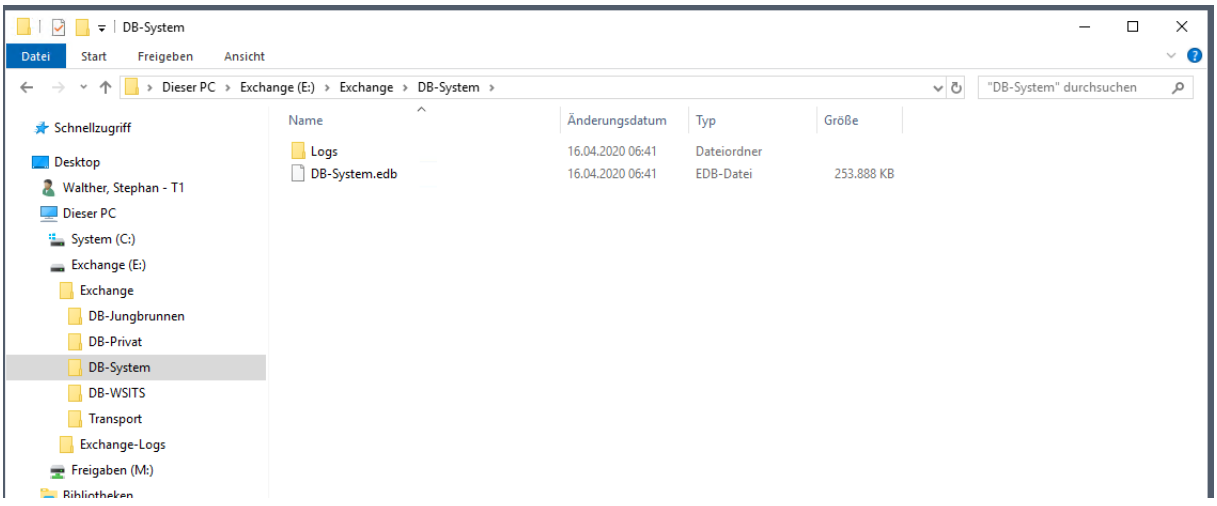
```
    # Konfiguration der neuen Datenbanken
        'WSITS','Jungbrunnen','Privat' |
            ForEach-Object {
                New-MailboxDatabase `
                        -Name          "DB-$_-neu" `
                        -EdbFilePath   "E:\Exchange\DB-$_\DB-$_.edb" `
                        -LogFolderPath "E:\Exchange\DB-$_\Logs" `
                        -Server        "WS-MX2"
                Mount-Database "DB-$_-neu"
            }
```

```
Name                    Server          Recovery        ReplicationType
----                    ------          --------        ---------------
DB-WSITS-neu            WS-MX2          False           None
WARNUNG: Starten Sie den Microsoft Exchange-Informationsspeicherdienst auf Server "WS-MX2" neu, nachdem Sie neue Postfachdatenba
nken hinzugefügt haben.
DB-Jungbrunnen-neu      WS-MX2          False           None
WARNUNG: Starten Sie den Microsoft Exchange-Informationsspeicherdienst auf Server "WS-MX2" neu, nachdem Sie neue Postfachdatenba
nken hinzugefügt haben.
DB-Privat-neu           WS-MX2          False           None
WARNUNG: Starten Sie den Microsoft Exchange-Informationsspeicherdienst auf Server "WS-MX2" neu, nachdem Sie neue Postfachdatenba
nken hinzugefügt haben.
```

Im Exchange Admin Center sind nun 8 Datenbanken enthalten:



Jede Datenbank hat diese Dateien auf der neuen Exchange-Partition. Auffällig ist bei Exchange Server 2019, dass der Indizierungsordner fehlt. Das ist eine lange ersehnte Verbesserung: Der Suchindex einer Datenbank ist nun IN der Datenbank enthalten und kann auch mit der Replikation in einer Verfügbarkeitsgruppe auf einen anderen Server repliziert werden. Vorher musste jeder Server die Datenbank lokal durchsuchen und indizieren!



Jetzt kommen noch ein paar Einstellungen auf alle neuen Datenbanken:

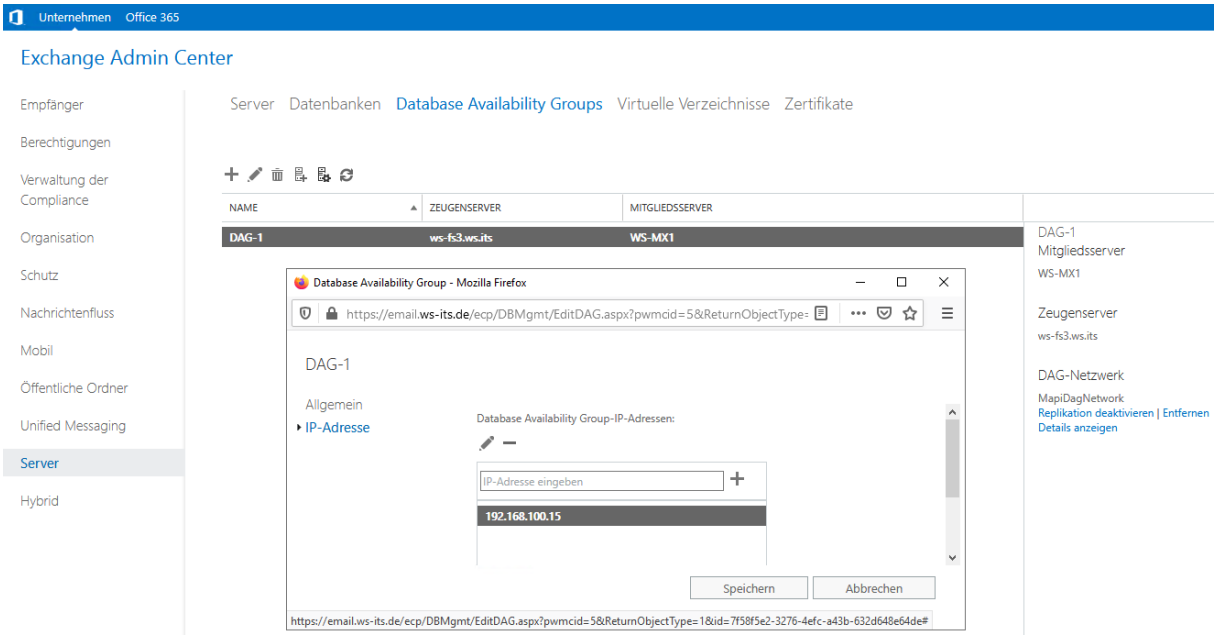
```
        Get-MailboxDatabase -Server "WS-MX2" |
            Set-MailboxDatabase `
                -RetainDeletedItemsUntilBackup $true `
                -DeletedItemRetention          65 `
                -MailboxRetention              30 `
                -ProhibitSendQuota             "unlimited" `
                -ProhibitSendReceiveQuota      "unlimited" `
                -OfflineAddressBook            (Get-OfflineAddressBook).name
```

Aufbau der neuen Datenbankverfügbarkeitsgruppe (DAG)

Eigentlich könnte ich jetzt die Mailboxen verschieben. Es fehlt aber ein wichtiges Element: Das Backup! Für die Datensicherung brauche ich eigentlich nicht viel. Ich möchte aber bei der Migration des anderen Mailservers nicht mehr viel anpassen müssen. Mein Backup kann mit der Datenbankverfügbarkeitsgruppe umgehen. Also sollte ich diese jetzt erstellen. Dann wird mein Backup-Programm später keine Probleme haben.

Eigentlich benötige ich für einen DAG-Cluster mehr als einen Server. Aber es ist nicht unmöglich.

Der alte Cluster hat eine Cluster-IP:



Den neuen Cluster erstelle ich ohne IP-Adresse mit einem anderen Namen. Den Zeugenserver kann ich aber weiterverwenden. Beim Anlegen der DAG erhalte ich eine Warnmeldung:

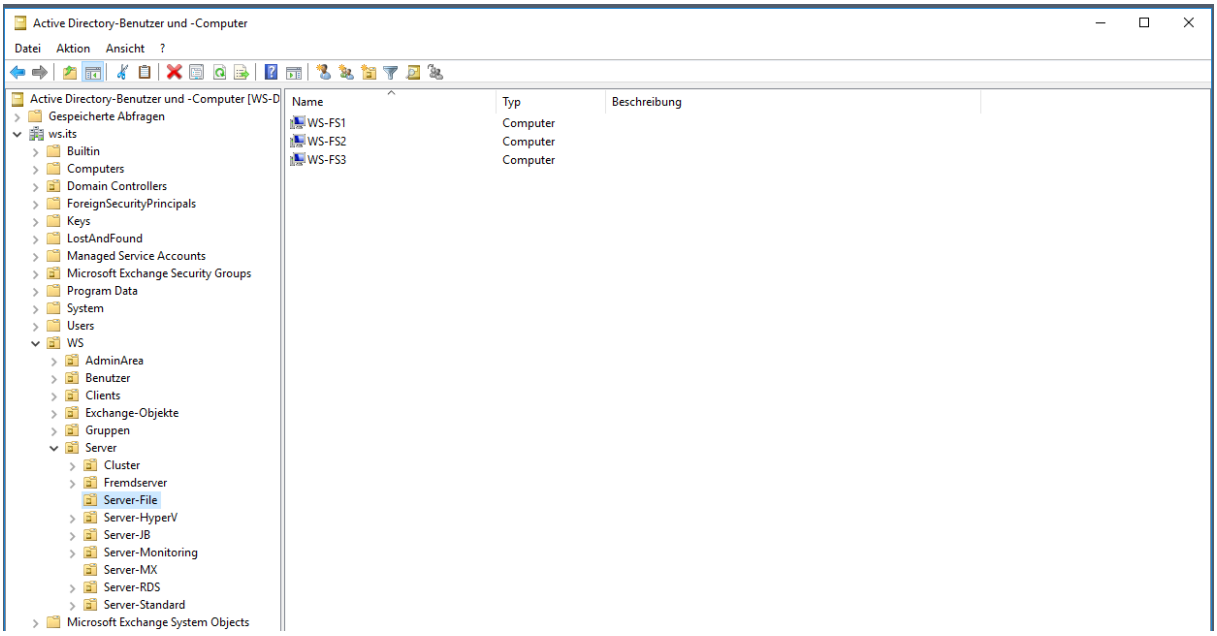
```
207     # Konfiguration der DAG
208         New-DatabaseAvailabilityGroup `
209                 -Name              "WS-DAG" `
210                 -WitnessServer     "WS-FS3.ws.its" `
211                 -WitnessDirectory "E:\DAG_Witness"
```

```
PS C:\> New-DatabaseAvailabilityGroup `
                -Name              "WS-DAG" `
                -WitnessServer     "WS-FS3.ws.its" `
                -WitnessDirectory "E:\DAG_Witness"
WARNUNG: Die Exchange-Gruppe "Vertrauenswürdiges Subsystem" ist kein Mitglied der lokalen Administratorengruppe auf dem angegebe
nen Zeugenserver "WS-FS3.ws.its". Fehler: Zugriff verweigert

Name            Member Servers                                  Operational Servers
----            --------------                                  -------------------
WS-DAG          {}
```

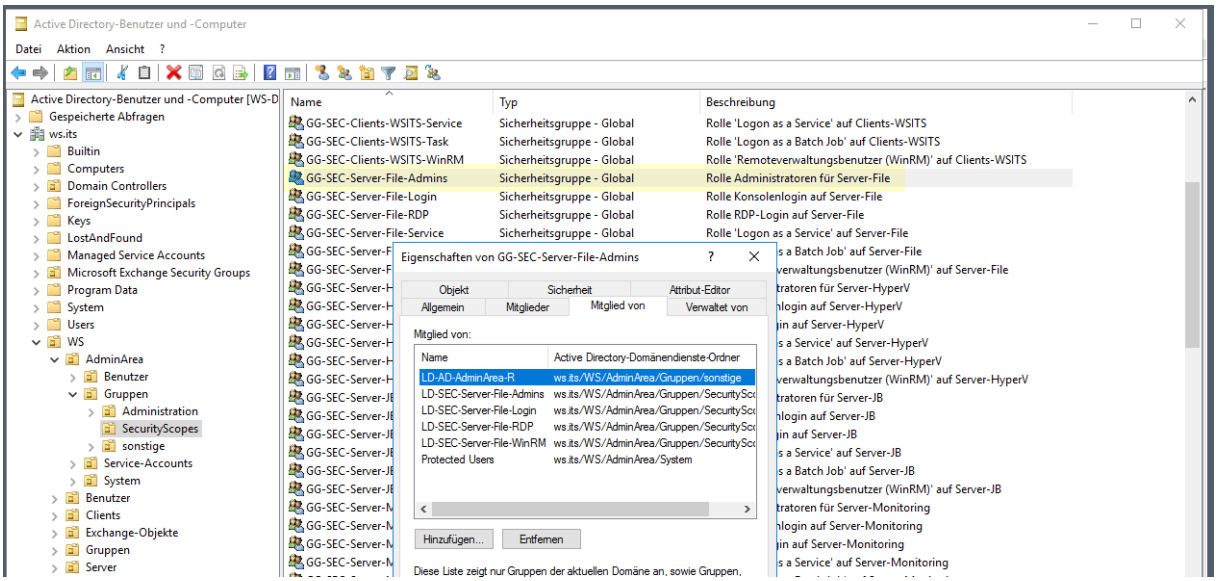
Ein Blick ins Active Directory zeigt an, dass die Gruppe tatsächlich kein Mitglied der lokalen Administratoren auf dem Fileserver WS-FS3 ist. Daher verschachtele ich sie in die richtige Gruppe.

Hintergrund:

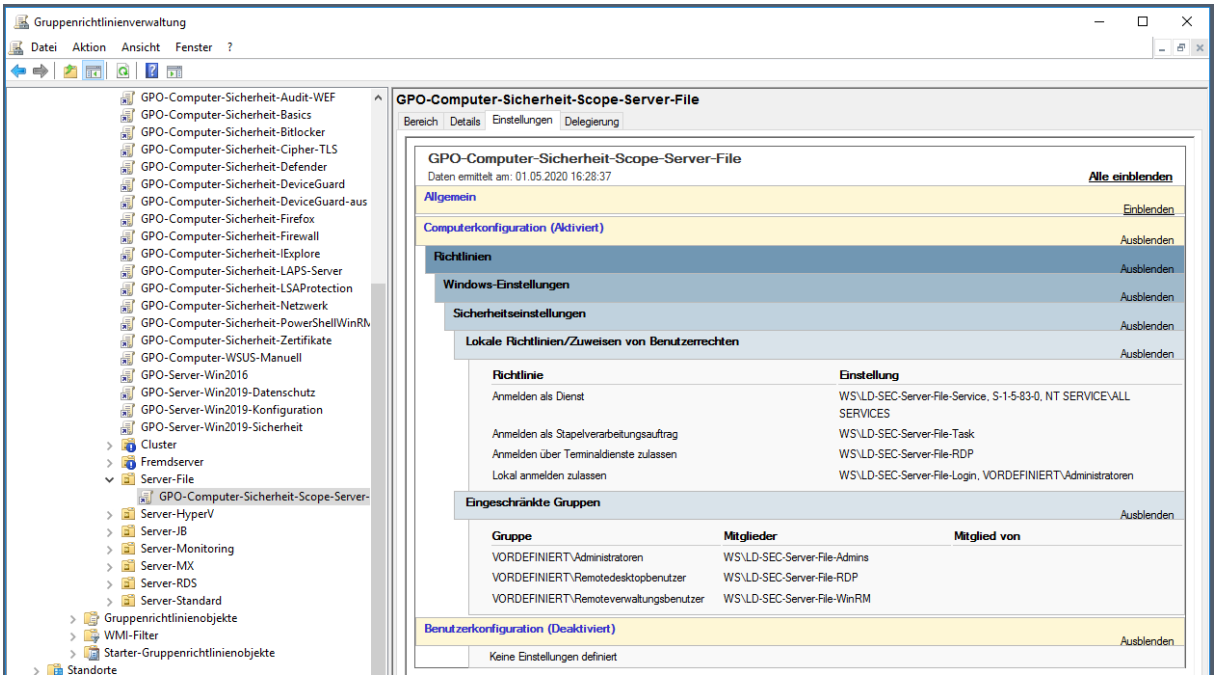
Ich habe meine Server aufgeteilt. Jede Servergruppe liegt in einer eigenen Organisationseinheit. Jede dieser OUs hat eine eigene Gruppenrichtlinie und eigene Gruppen im Active Directory. Die Gruppenrichtlinie setzt nun für die AD-Gruppen die lokalen Rechte auf den Servern um. So kann ich z.B. die Berechtigung „lokal administrative Rechte“ für eine Menge von Servern explizit im Active Directory steuern. Hier sieht man die OU mit den Servern:



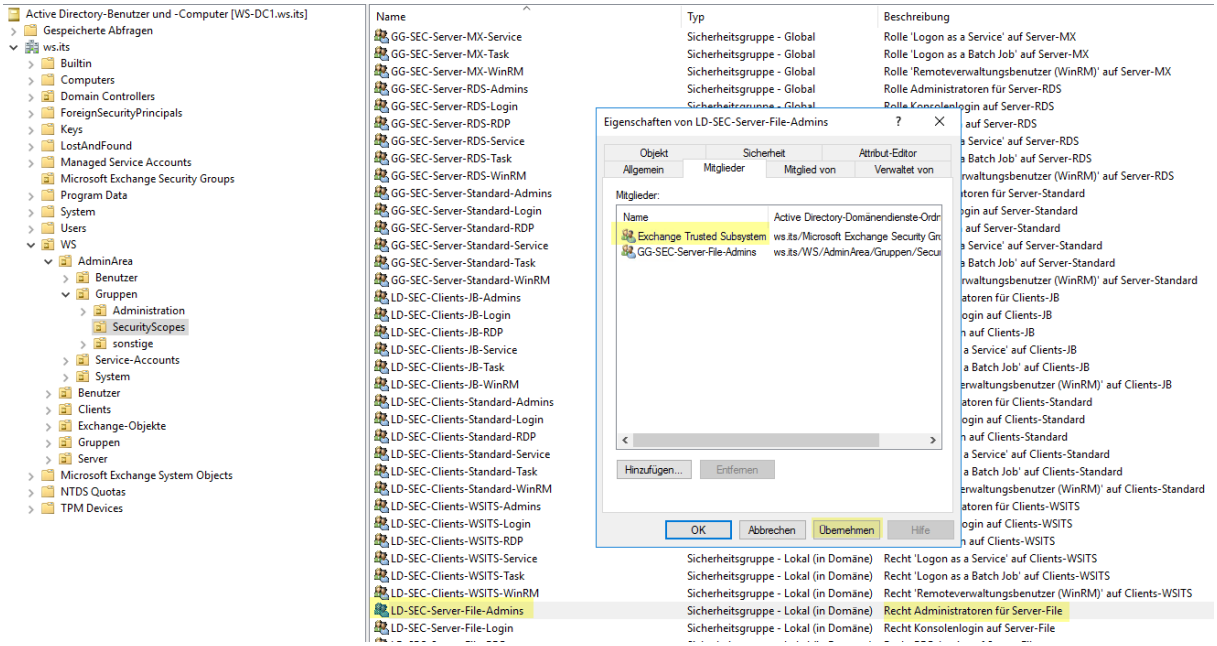
Das sind die administrativen Gruppen:



Und hier die dazugehörige Gruppenrichtlinie:

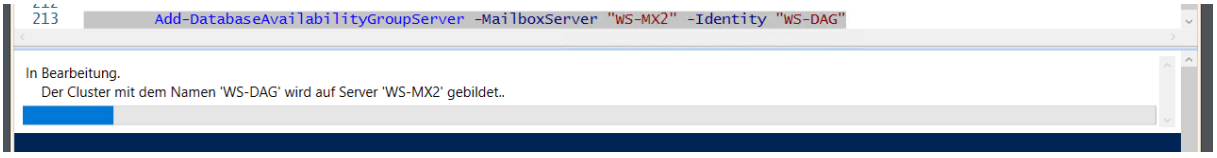


Mitglieder der Gruppe „LD-SEC-Server-File-Admins“ sind also auch Mitglieder der Gruppe „Administratoren“ auf meinen Fileservern. Also kann ich hier auch die erforderliche Gruppe „Exchange Trusted Subsystems“ aufnehmen:

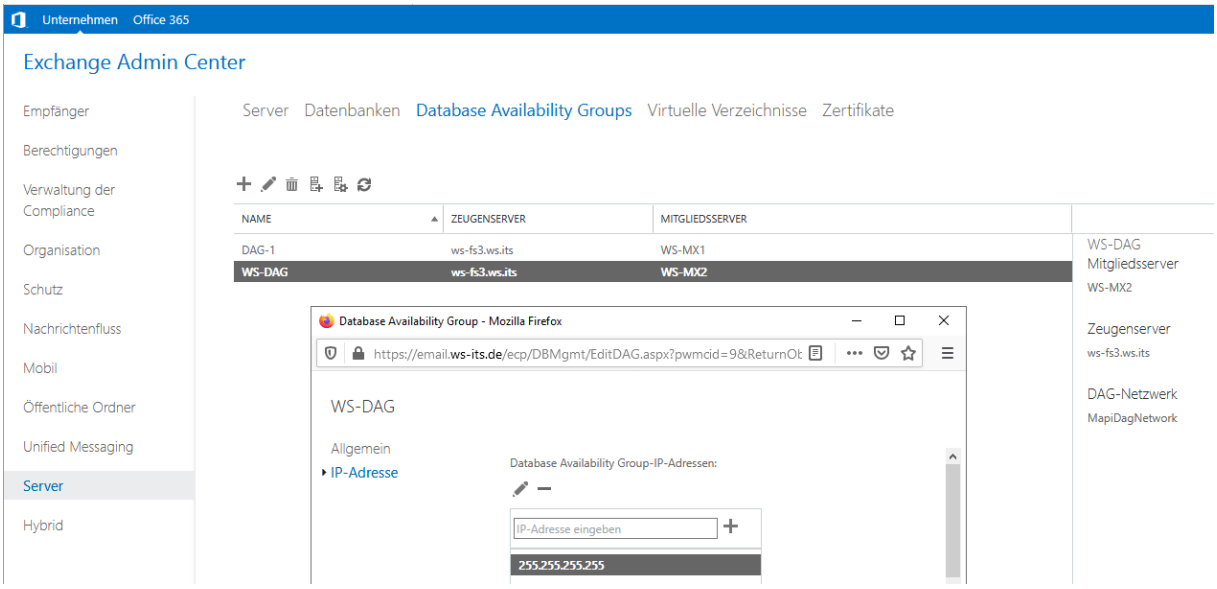


Mit diesem Modell der Berechtigung habe ich unter anderem auch den Vorteil, dass ich lokale Rechte ausschließlich über das Active Directory steuern kann.

Weiter geht es mit dem Aufbau der DAG. Bisher ist sie nur ein Datensatz im Active Directory. Erst mit dem ersten Mitglied wird sie erstellt:

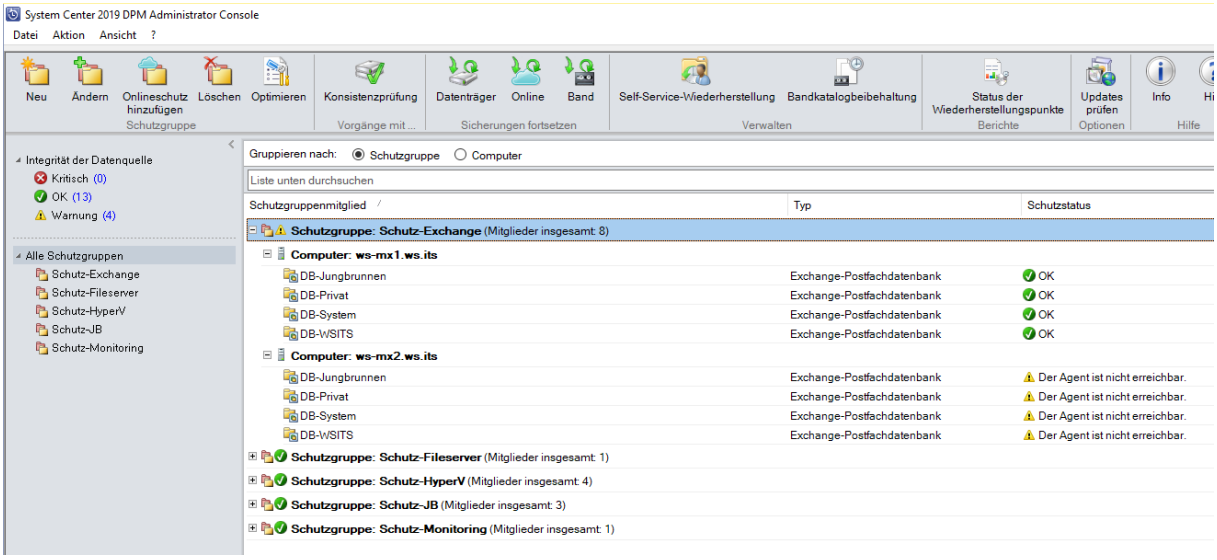


Mit der im Hintergrund replizierten Gruppenmitgliedschaft konnte die neue DAG erstellt werden. Ohne eine Cluster-IP wird die Broadcast-Adresse als Platzhalter im EAC angezeigt:

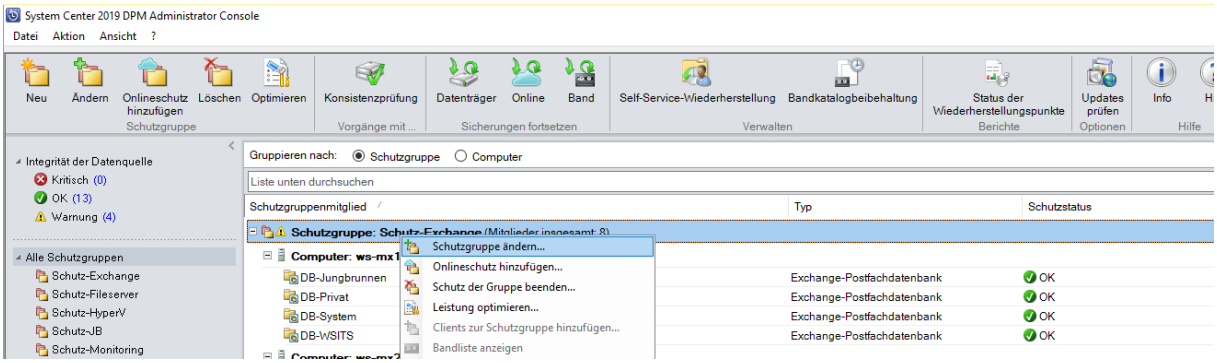


Konfiguration der Datensicherung mit dem DPM 2019

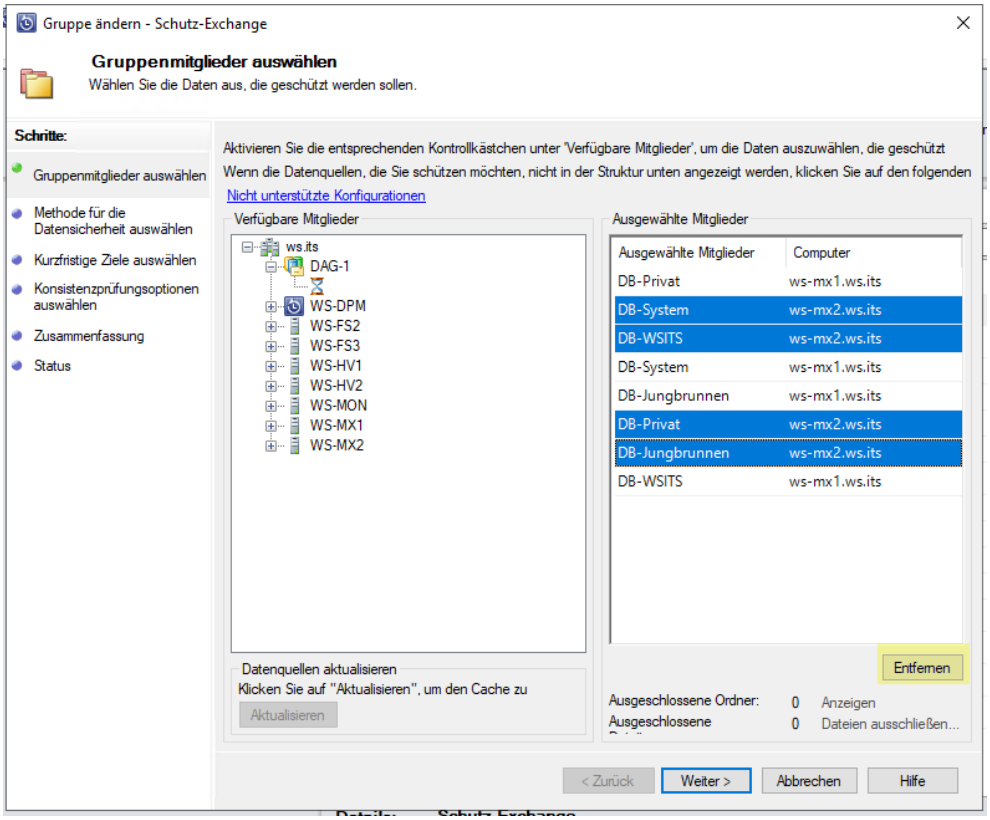
Jetzt kann ich für die neue DAG im Data Protection Manager eine Datensicherung planen. Vorher muss ich aber noch die alte Sicherungskonfiguration bereinigen. Denn hier steht noch der alte Server drin:



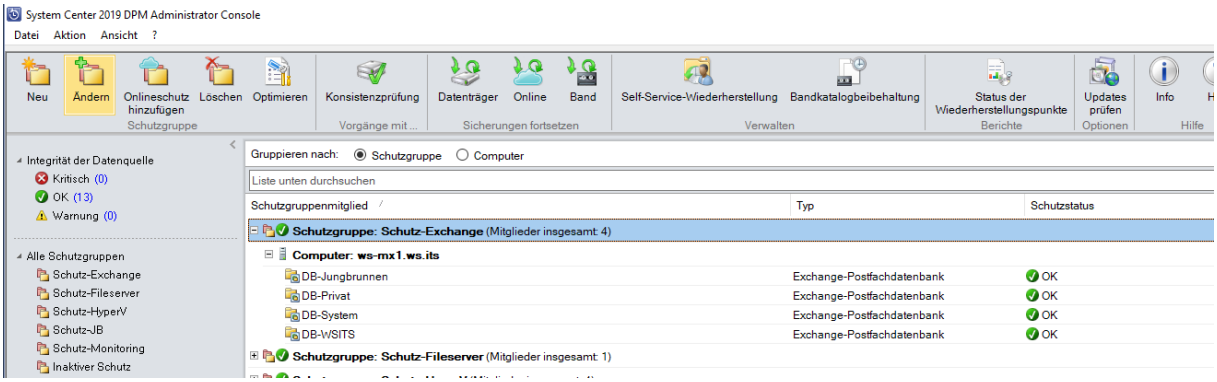
Ich verändere die Schutzgruppe:



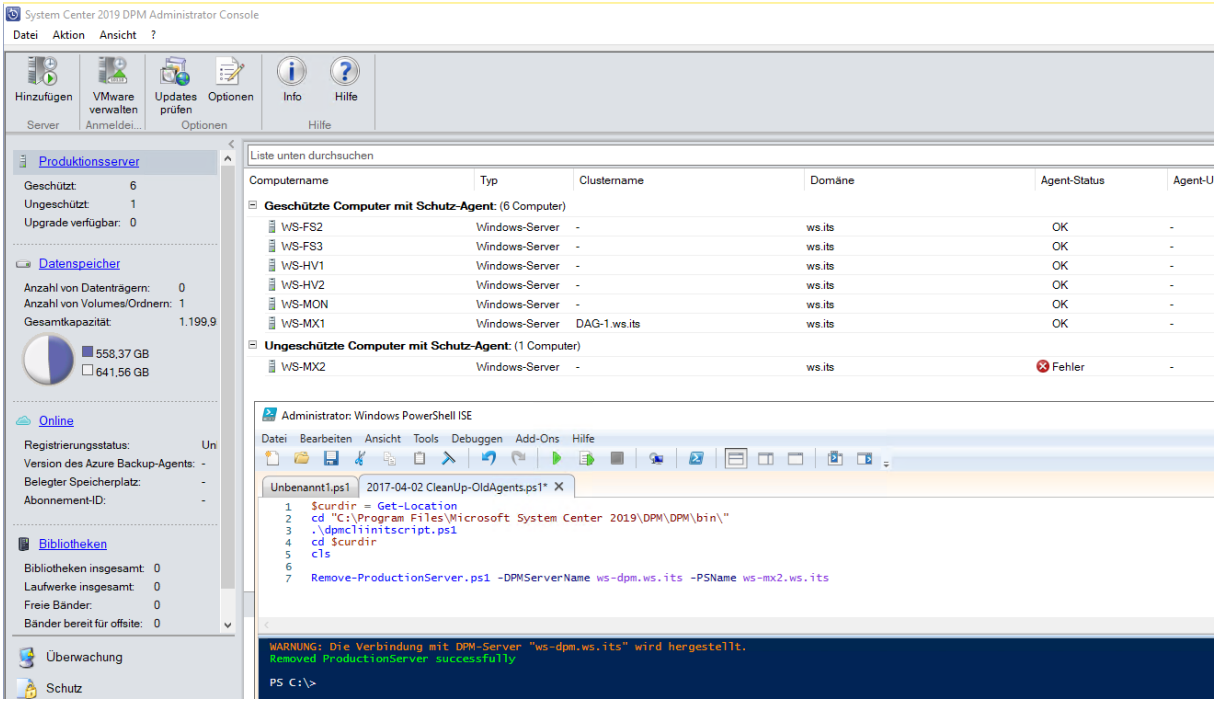
Die alten Sicherungseinträge entferne ich einfach:



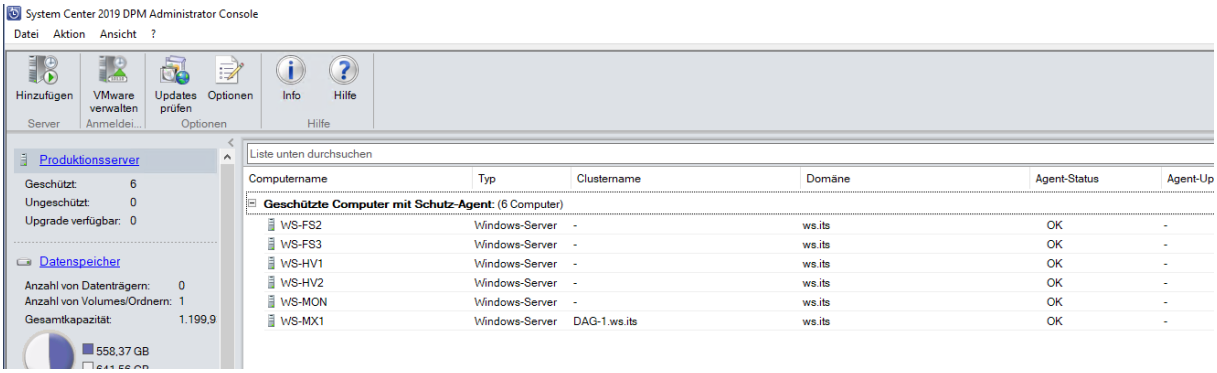
Ein paar Klicks später ist die alte Schutzgruppe sauber:



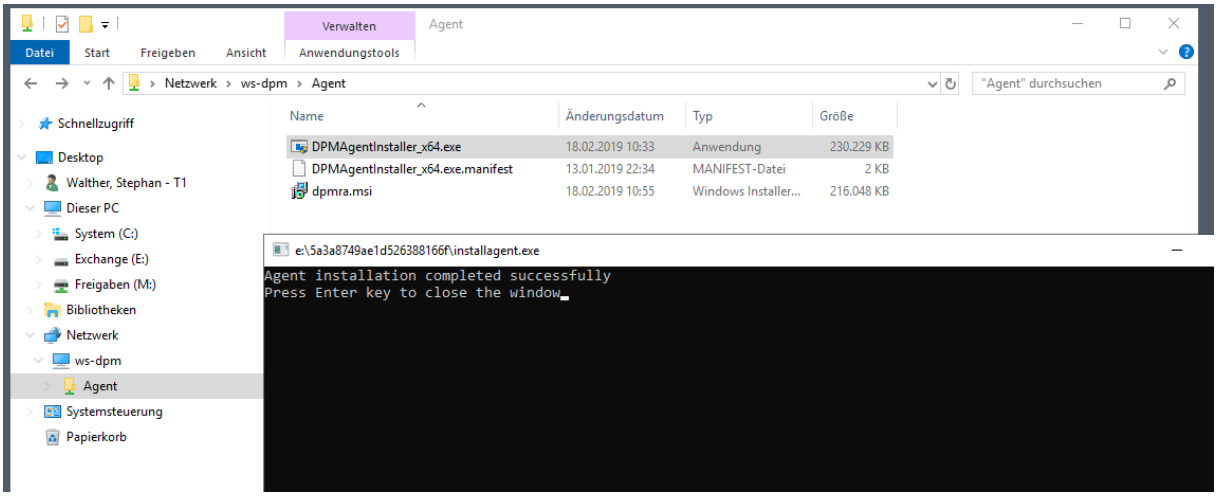
Nun muss ich noch die alte Agent-Verbindung zum nicht mehr vorhandenen Windows Server 2016 WS-MX2 entfernen. Das geht leider nicht mit der grafischen Oberfläche. Daher verwende ich einen PowerShell-Befehl:



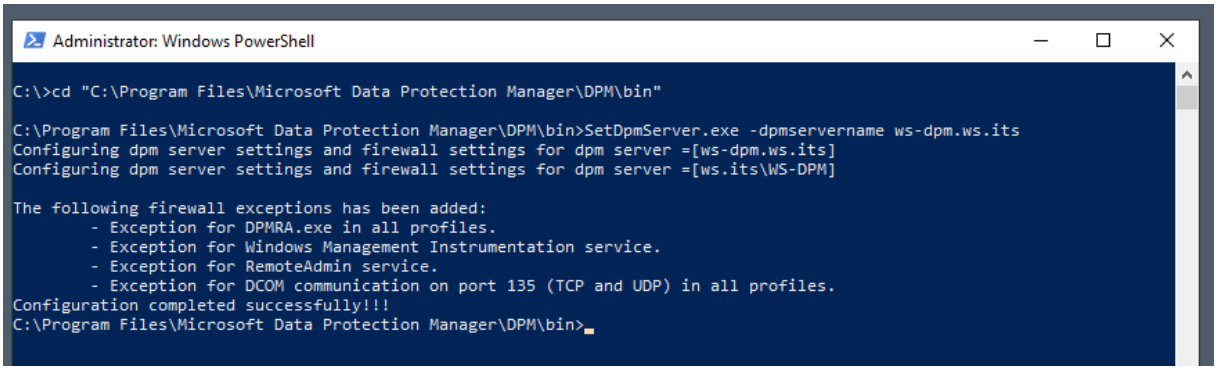
Jetzt hat der DPM den alten Mailserver vergessen:



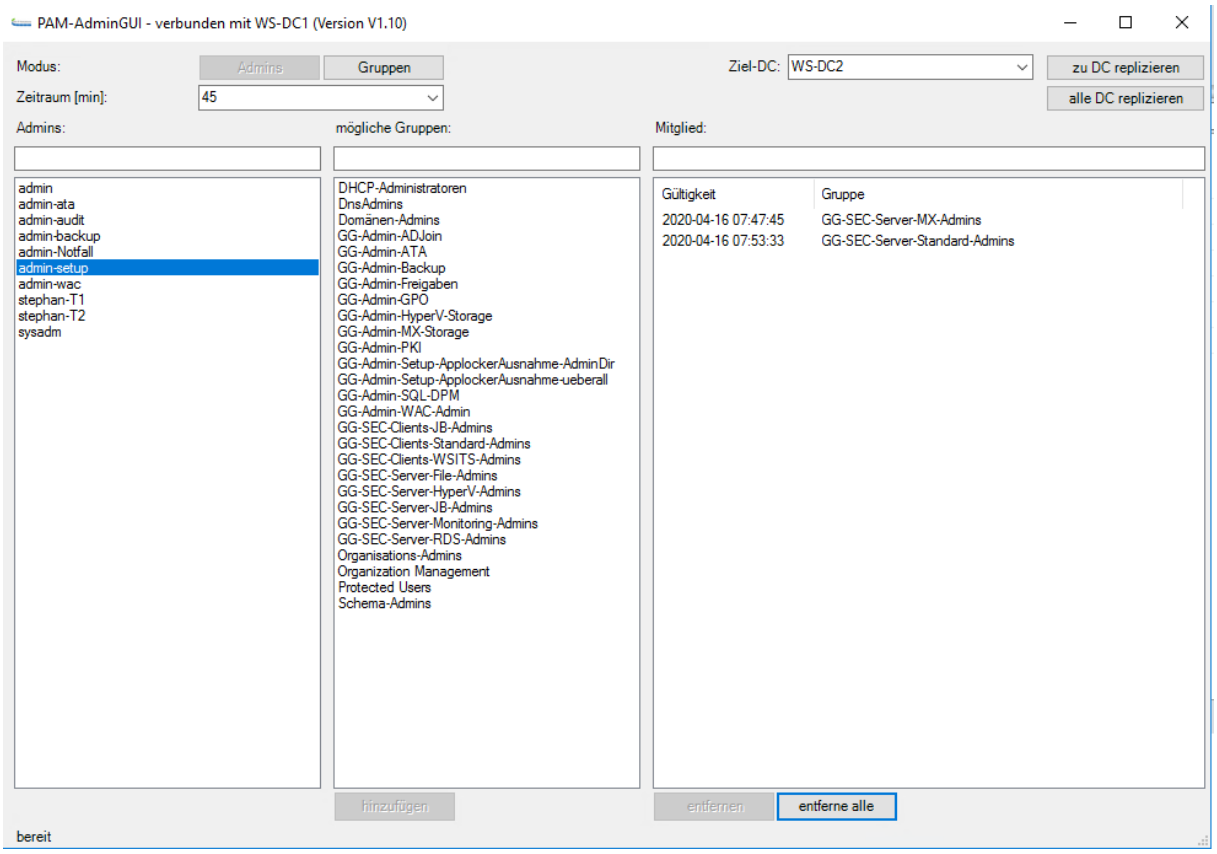
Auf dem neuen Mailserver installiere ich den Agent des DPM. Das geht lokal einfach am besten. Auf meinem DPM hatte ich dazu eine Freigabe erstellt:



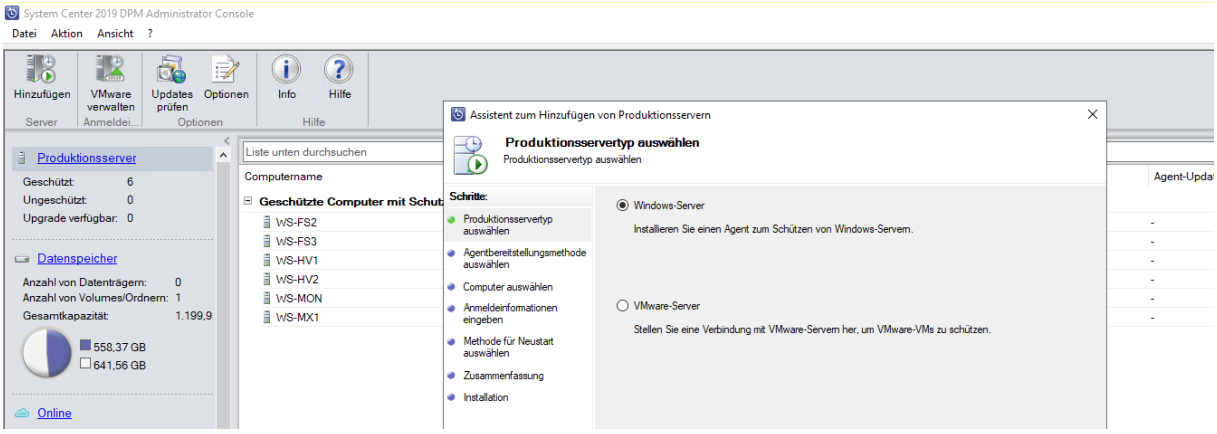
Nach der Installation muss der Agent auf seinen neuen DPM-Server vorbereitet werden. Dabei werden Firewall-Ausnahmen erstellt:



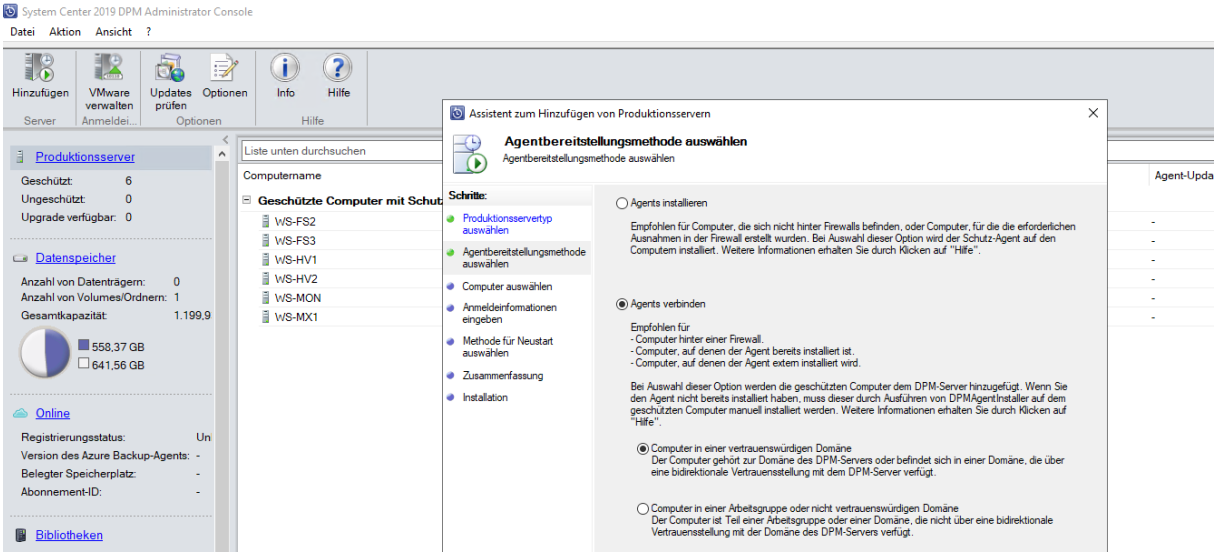
Für die Verbindung des DPM-Agents mit dem DPM-Server benötige ich eine Kennung, die administrative Rechte auf dem DPM-Server (Standard) und den Exchange Servern (MX) hat. Durch meine administrativen Trennungen mit den GPOs gibt es einen solchen Account aktuell nicht. Er wird aber nur kurz benötigt. Zudem darf er kein Mitglied der Gruppe „Protected Users“ sein. Für solche Fälle habe ich den Account „admin-setup“. Dieser hat keine statischen Gruppenmitgliedschaften. Mit meinem PAM-PowerShell-Script weise ich die zeitlich begrenzten Mitgliedschaften zu:



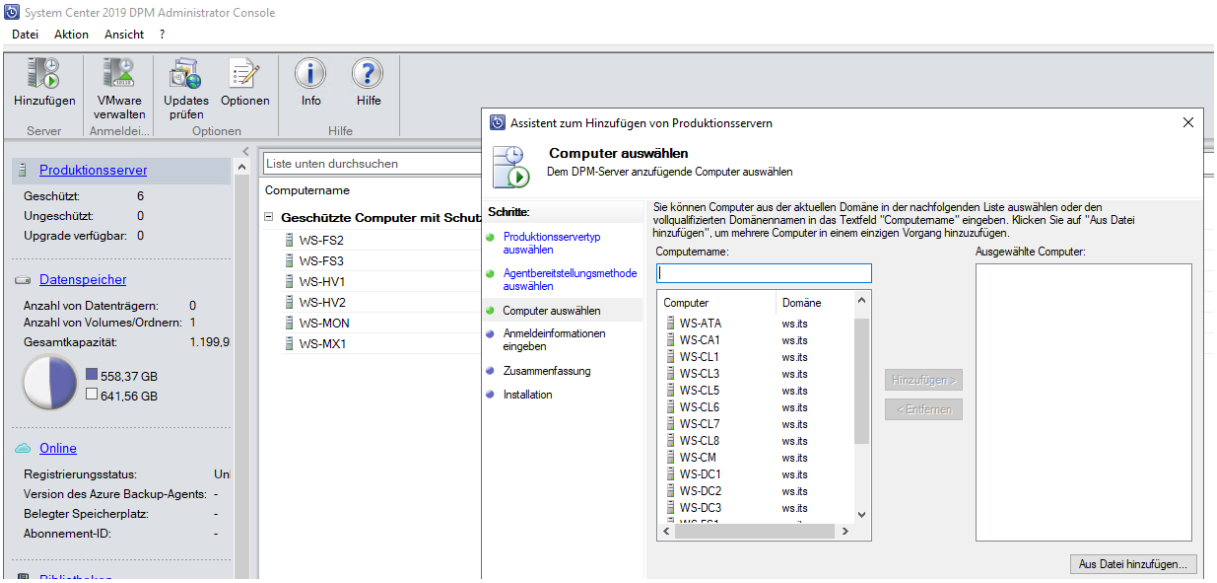
Weiter geht es im DPM in der Verwaltungsseite der Management-Konsole. Hier kann ich einen neuen Agent mit dem Schalter „Hinzufügen Server“ einrichten:



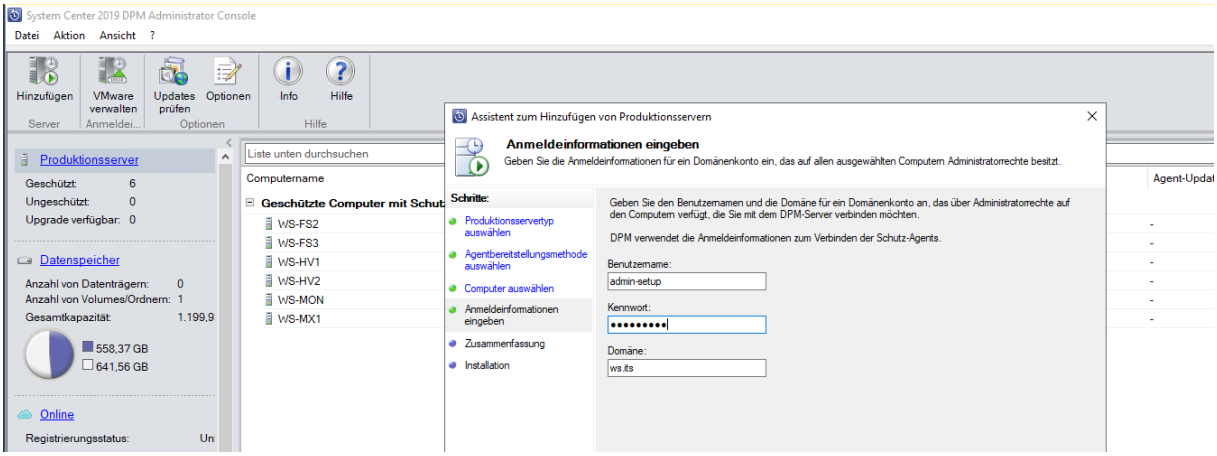
Wichtig ist, dass nur noch eine Verbindung aufgebaut werden muss. Der Agent ist ja bereits installiert:



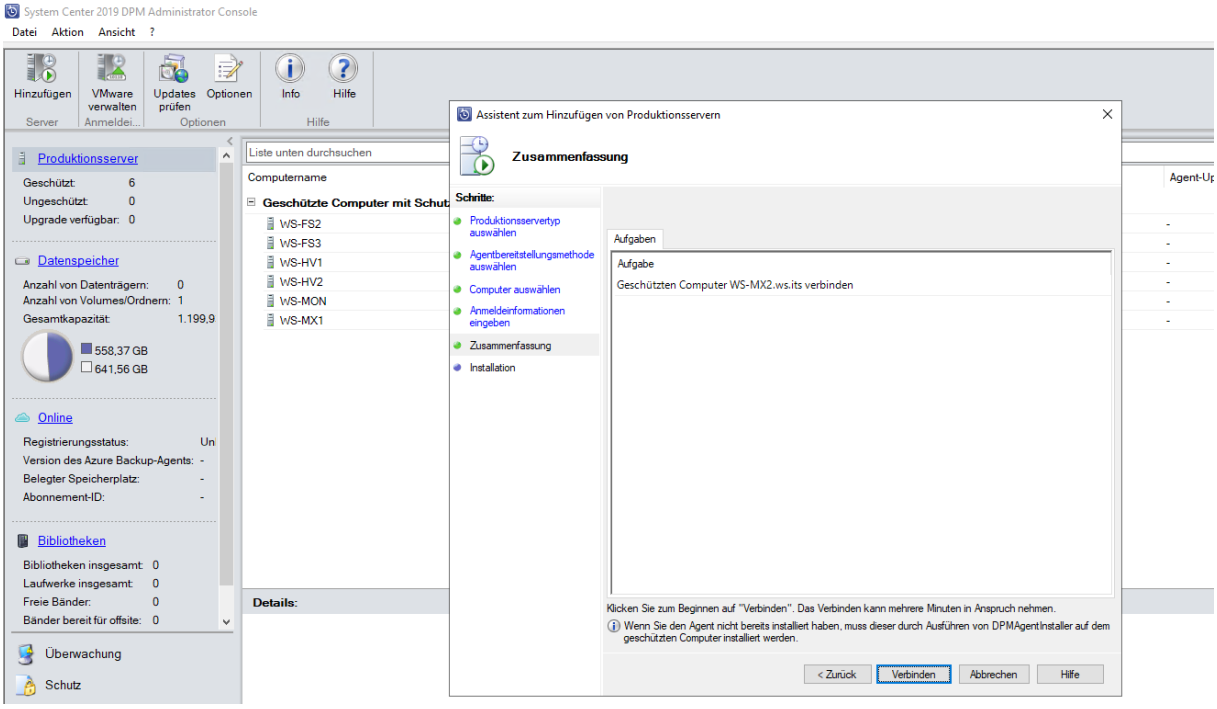
Jetzt suche ich den Server aus der Liste aus. Nach dem Hinzufügen wird er nicht mehr angezeigt. Das ist ein Bug:



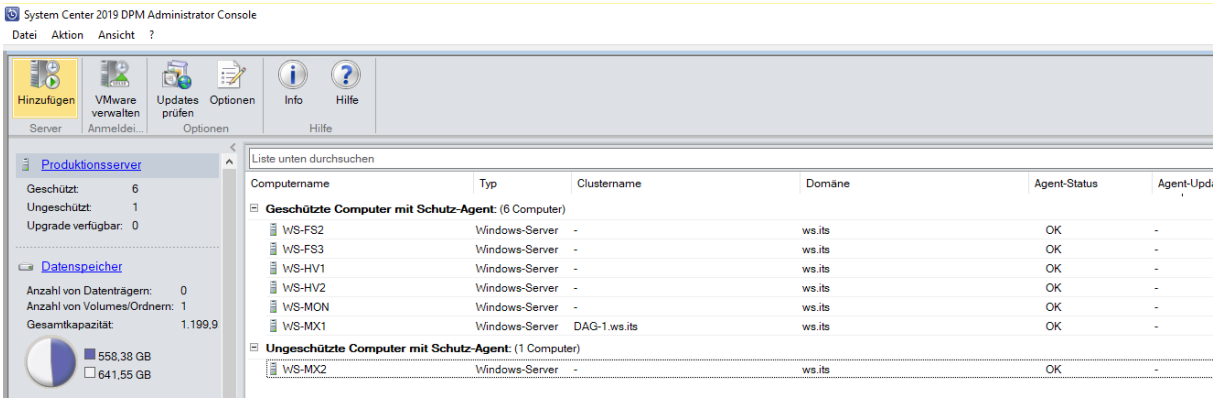
Anschließend gebe ich die Anmeldeinformationen des zuvor konfigurieren Admin-Accounts ein:



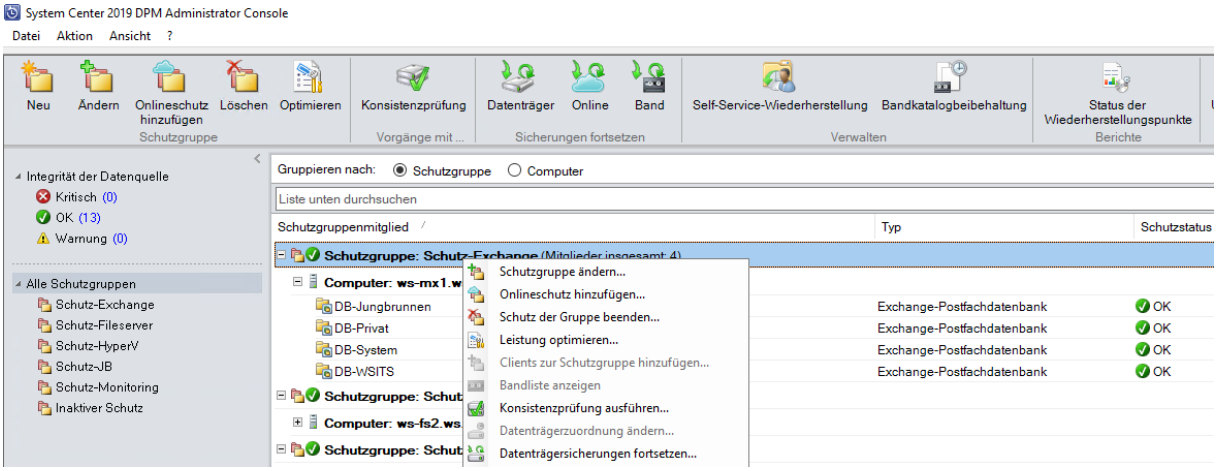
Und dann kann der DPM beginnen:



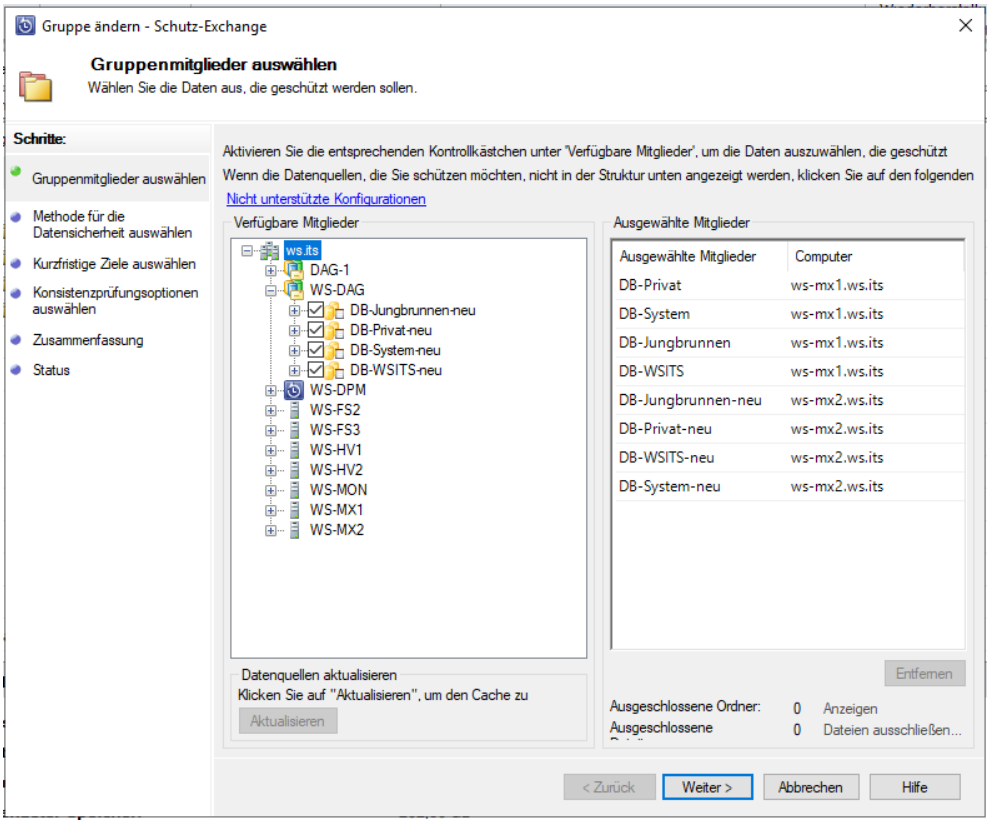
Der neue Mailserver wird angezeigt:



Weiter geht es mit der Konfiguration der Datensicherung. Ich möchte die Definitionen der aktuellen Schutzgruppe weiterverwenden. Also starte ich die Änderung:



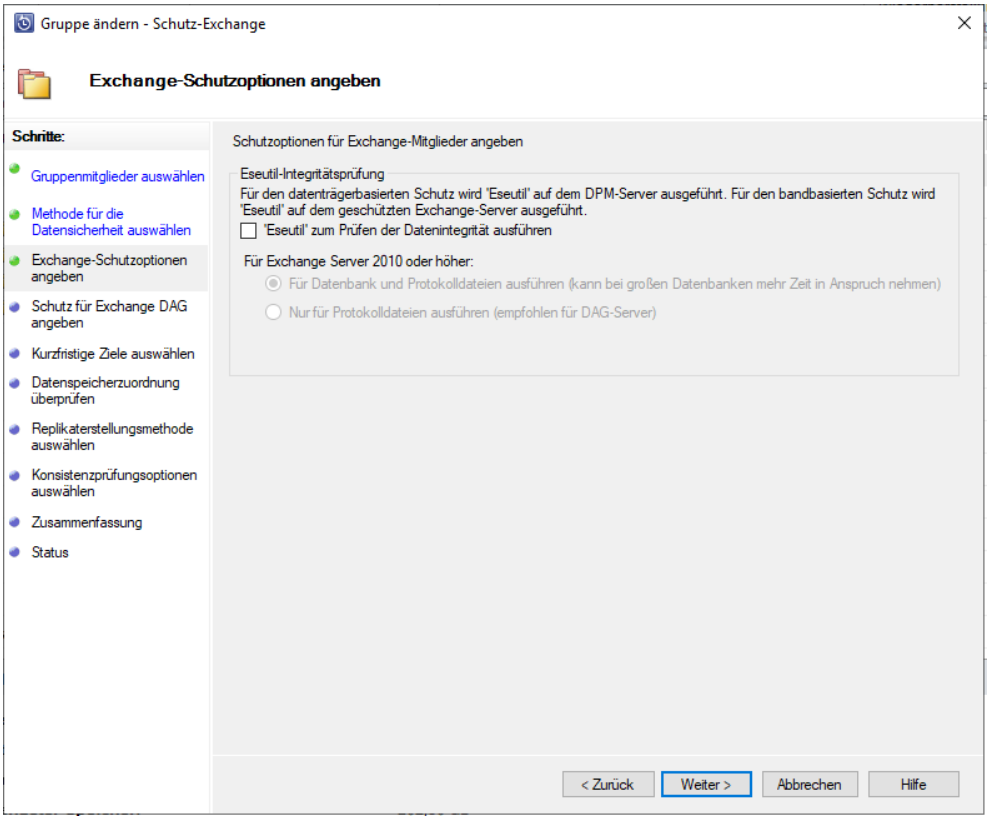
Zu der alten DAG wähle ich die 4 neuen Datenbanken der neuen DAG aus:



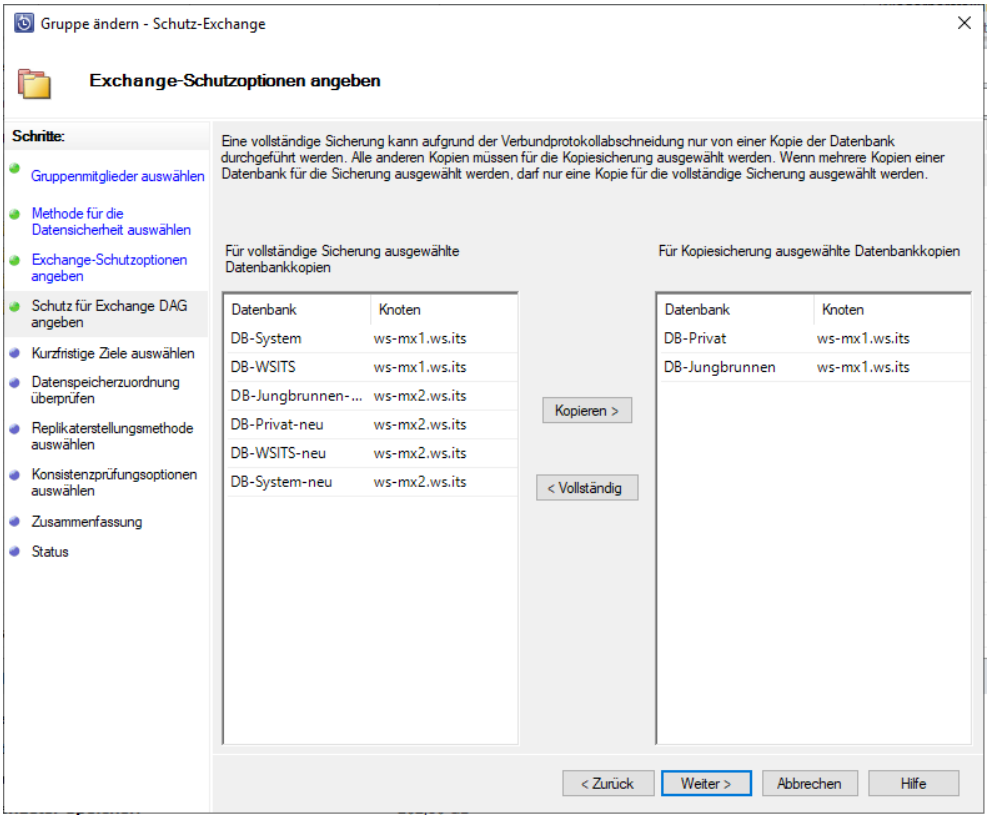
Die Überprüfung mit eseutil muss ich an dieser Stelle herausnehmen. Ein DPM benötigt dazu explizit die passende eseutil.exe einer Exchange Server Version. Ich habe aber gerade eine Mischumgebung. Hier komme ich nach Abschluss der Migration noch einmal zurück.

Hintergrund:

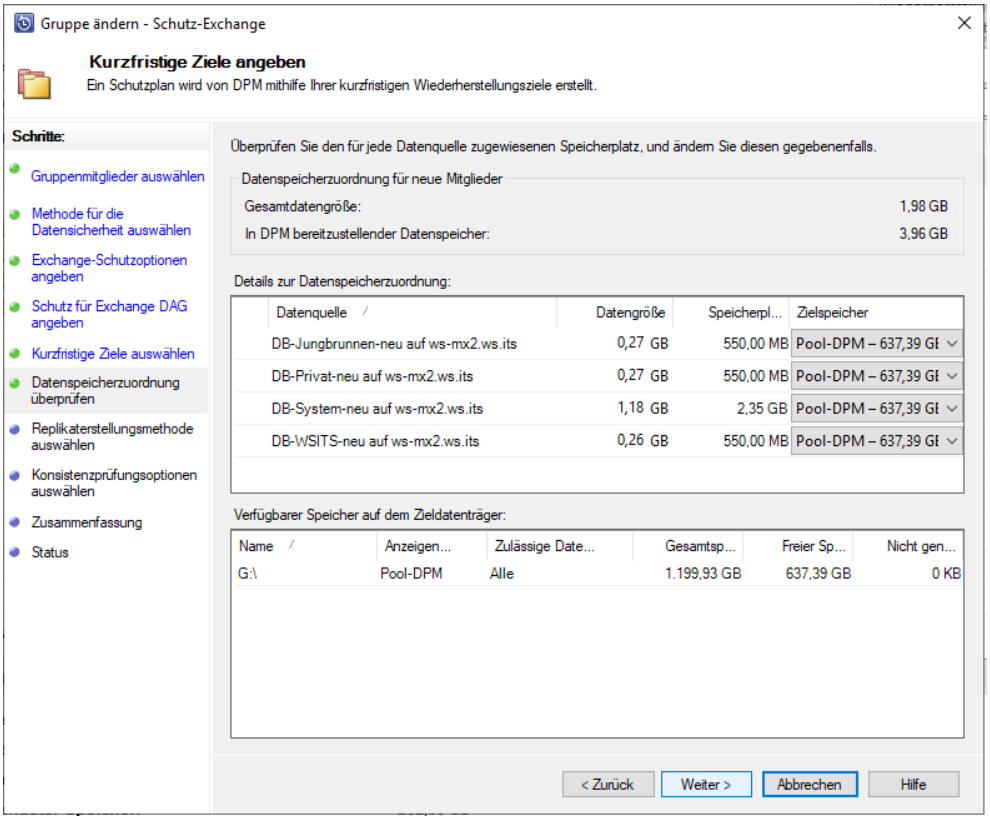
Die Datenbanken werden im laufenden Betrieb mittels VSS-Snapshot gesichert. Sie wurden also nicht korrekt heruntergefahren. Bei der Sicherung ist das auch kein Problem. Das tritt erst bei einer Wiederherstellung auf. Dabei wird die gesicherte edb-Datenbankdatei aus dem DPM extrahiert und im Exchange Server gespeichert. Der Information Store Service kann aber nur korrekt heruntergefahrene Datenbanken mounten. Die gesicherte Datei ist aber in einem „Dirty-Shutdown“-Zustand. Mit eseutil kann die Datenbank „repariert“ bzw. in einen „Clean-Shutdown“-State gebracht werden. Leider kostet diese Arbeit neben Fachwissen vor allem Zeit. Und die haben wir nicht im Recovery-Fall. Daher kann der DPM jede gesicherte Datenbank automatisch in einen „Clean-Shutdown“ überführen. Bei einer Recovery wird sie einfach extrahiert und kann anschließend direkt gemountet werden.



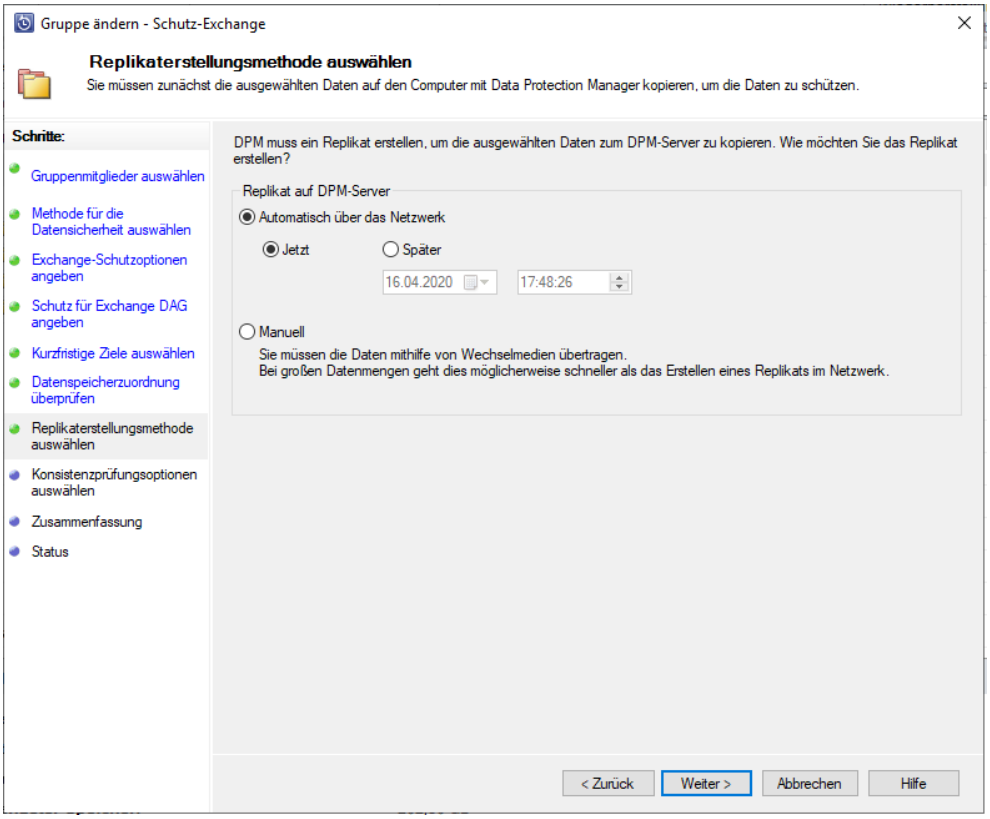
Die 4 Datenbanken sollen mit einem Full-Backup gesichert werden:



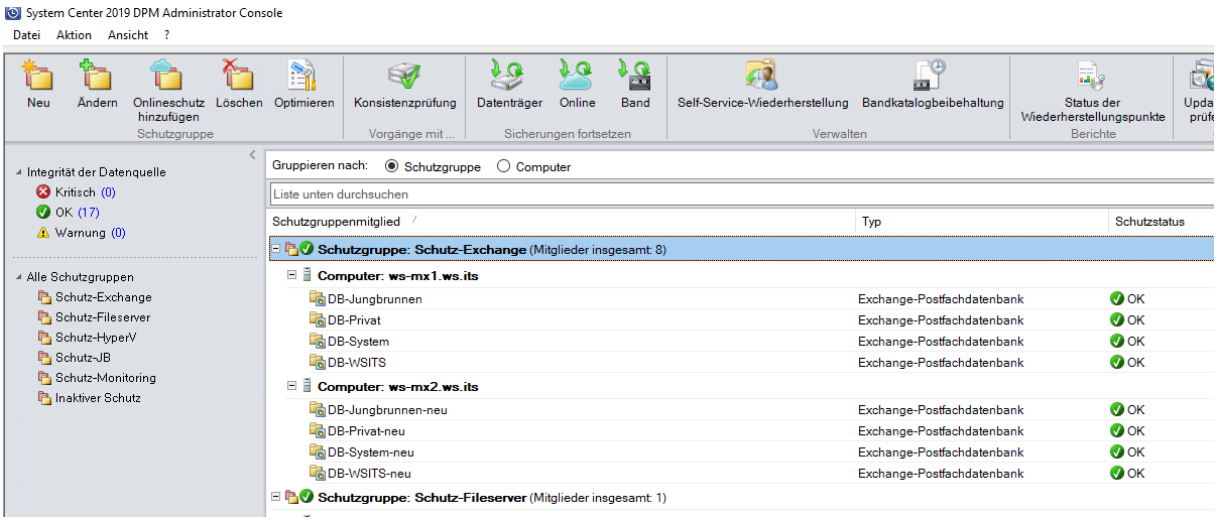
Der Speicherplatz wird in einem Pool organisiert:



Die Sicherung darf sofort starten. Noch sind die Datenbanken klein. Das sollte nicht lange dauern:



Und nach wenigen Minuten sind die leeren Datenbanken gesichert:



Verschiebung der Mailboxen

Endlich kann ich meine Mailboxen aus den gesicherten, alten Mailboxdatenbanken in die gesicherten, neuen DBs verschieben. Die Postfächer sind sauber in den Datenbanken organisiert. Ich verschaffe mir einen Überblick. Dabei sollen auch die System-Mailboxen Beachtung finden:

```
215     # Migration der Mailboxen
216         $Mailboxen = @()
217         $Mailboxen += Get-Mailbox
218         $Mailboxen += Get-Mailbox -Arbitration
219         $Mailboxen += Get-Mailbox -PublicFolder
220         $Mailboxen += Get-Mailbox -AuditLog
221         $Mailboxen += Get-Mailbox -GroupMailbox
222
223         $Mailboxen | Format-Table -Property alias,database,ArchiveDatabase
224
```

```
Alias                                                Database       ArchiveDatabase
-----                                                --------       ---------------
stephan.walther                                      DB-WSITS       DB-WSITS
Administrator                                        DB-System
Nicole                                               DB-Jungbrunnen
Sabine                                               DB-Jungbrunnen
Sandro                                               DB-Privat
Romy                                                 DB-Privat
DiscoverySearchMailbox{D919BA05-46A6-415f-80AD-7E09334BB852} DB-System
stephan-jb                                           DB-Jungbrunnen DB-Jungbrunnen
stephan-privat                                       DB-Privat
Marketing                                            DB-Jungbrunnen
Jungbunnen                                           DB-Jungbrunnen
Stephan-AD                                           DB-System
Technik                                              DB-WSITS
SystemMailbox{bb558c35-97f1-4cb9-8ff7-d53741dc928c}  DB-System
SystemMailbox{D0E409A0-AF9B-4720-92FE-AAC869B0D201}  DB-System
SystemMailbox{e0dc1c29-89c3-4034-b678-e6c29d823ed9}  DB-System
FederatedEmail.4c1f4d8b-8179-4148-93bf-00a95fa1e042  DB-System
Migration.8f3e7716-2011-43e4-96b1-aba62d229136       DB-System
SystemMailbox{1f05a927-e44d-4543-8a6e-7145df37ed60}  DB-System
SystemMailbox{2CE34405-31BE-455D-89D7-A7C7DA7A0DAA}  DB-System-neu
PF-Technik                                           DB-WSITS
PF-Jungbrunnen                                       DB-Jungbrunnen
SystemMailbox{8cc370d3-822a-4ab8-a926-bb94bd0641a9}  DB-System-neu
```

Die Verschiebungen plane ich mit ein paar PowerShell-Zeilen. Von jeder Mailbox lese ich den Namen der alten DB aus un d verschiebe sie in die DB mit dem alten Namen plus dem Suffix „-neu“. Archive werden dabei ebenfalls berücksichtigt. Damit es besonders schnell geht, weise ich die Verschiebung mit der Priorität „emergency“ an. Es gibt schließlich bald Abendessen!

```
$Mailboxen |
    Where-Object { $_.database -notlike '*neu' } |
        ForEach-Object {
            if ($_.ArchiveDatabase -ne $null) {
                New-MoveRequest `
                        -Identity $_.alias `
                        -TargetDatabase ($_.Database + "-neu") `
                        -Priority emergency `
                        -ArchiveTargetDatabase ($_.ArchiveDatabase + "-neu")
            } else {
                New-MoveRequest `
                        -Identity $_.alias `
                        -TargetDatabase ($_.Database + "-neu") `
                        -Priority emergency
            }
        }
```

```
}

DisplayName      StatusDetail        TotalMailboxSize              TotalArchiveSize              PercentComplete
-----------      ------------        ----------------              ----------------              ---------------
Walther, Stephan WaitingForJobPickup 1.34 GB (1,439,248,827 bytes) 2.126 GB (2,283,185,655 bytes) 0
Administrator    WaitingForJobPickup 2.073 MB (2,174,122 bytes)                                   0
Widmann, Nicole  WaitingForJobPickup 541.5 MB (567,778,143 bytes)                                 0
Kroll, Sabine    WaitingForJobPickup 271.1 MB (284,218,269 bytes)                                 0
Widmann, Sandro  WaitingForJobPickup 153.6 MB (161,037,628 bytes)                                 0
Heyne, Romy      WaitingForJobPickup 48.77 MB (51,134,452 bytes)                                  0
DiscoverySear... WaitingForJobPickup 61.95 KB (63,438 bytes)                                      0
Walther, Step... WaitingForJobPickup 234.3 MB (245,647,090 bytes)  532.3 MB (558,145,692 bytes)   0
Walther, Step... WaitingForJobPickup 383.6 MB (402,197,079 bytes)                                 0
Marketing - J... WaitingForJobPickup 33.93 MB (35,579,055 bytes)                                  0
Jungbrunnen N... WaitingForJobPickup 187.2 MB (196,311,198 bytes)                                 0
Walther, Step... WaitingForJobPickup 528.4 KB (541,035 bytes)                                     0
Technik          WaitingForJobPickup 116.6 MB (122,227,085 bytes)                                 0
Microsoft Exc... WaitingForJobPickup 8.277 MB (8,678,874 bytes)                                   0
Microsoft Exc... WaitingForJobPickup 1.932 MB (2,026,274 bytes)                                   0
Microsoft Exc... WaitingForJobPickup 164.1 KB (168,078 bytes)                                     0
Microsoft Exc... WaitingForJobPickup 59.08 KB (60,500 bytes)                                      0
Microsoft Exc... WaitingForJobPickup 67.53 KB (69,149 bytes)                                      0
Microsoft Exc... WaitingForJobPickup 5.283 MB (5,539,852 bytes)                                   0
PF-Technik       WaitingForJobPickup 1.232 GB (1,323,300,586 by...                                0
PF-Jungbrunnen   WaitingForJobPickup 97.34 KB (99,674 bytes)                                      0
```

Mit „emergency“ wandern die Mailboxen in Windeseile auf den neuen Server:

```
Get-MoveRequest
```

```
DisplayName                                                     Status    TargetDatabase
-----------                                                     ------    --------------
Walther, Stephan                                                Completed DB-WSITS-neu
Administrator                                                   Completed DB-System-neu
Microsoft Exchange                                              Completed DB-System-neu
Widmann, Nicole                                                 Completed DB-Jungbrunnen-neu
Widmann, Sandro                                                 Completed DB-Privat-neu
Heyne, Romy                                                     Completed DB-Privat-neu
PF-Technik                                                      Completed DB-WSITS-neu
Walther, Stephan - Jungbrunnen Neufahrn                         Completed DB-Jungbrunnen-neu
Walther, Stephan - Privat                                       Completed DB-Privat-neu
Kroll, Sabine                                                   Completed DB-Jungbrunnen-neu
Walther, Stephan - T1                                           Completed DB-System-neu
Marketing - Jungbrunnen Neufahrn                                Completed DB-Jungbrunnen-neu
Jungbrunnen Neufahrn                                            Completed DB-Jungbrunnen-neu
DiscoverySearchMailbox {D919BA05-46A6-415f-80AD-7E09334BB852}   Completed DB-System-neu
Microsoft Exchange                                              Completed DB-System-neu
Microsoft Exchange Migration                                    Completed DB-System-neu
PF-Jungbrunnen                                                  Completed DB-Jungbrunnen-neu
Microsoft Exchange                                              Completed DB-System-neu
Microsoft Exchange-Genehmigungs-Assistent                       Completed DB-System-neu
Microsoft Exchange-Genehmigungs-Assistent                       Completed DB-System-neu
Technik                                                         Completed DB-WSITS-neu
```

Abschließend entferne ich die Verschiebeanforderungen:

```
Get-MoveRequest | Remove-MoveRequest
```

Damit ist auch die dritte Rolle produktiv. Die Mailboxbenutzer haben von den Verschiebungen übrigens nichts mitbekommen. Verbindungen von den Clients werden immer zum CAS-Server aufgebaut. Nur dieser muss die neue Location eines Mailbox-Elementes im Hintergrund lernen. Der Vorgang ist vollkommen transparent!

Nacharbeiten

Lizensierung des Exchange Servers

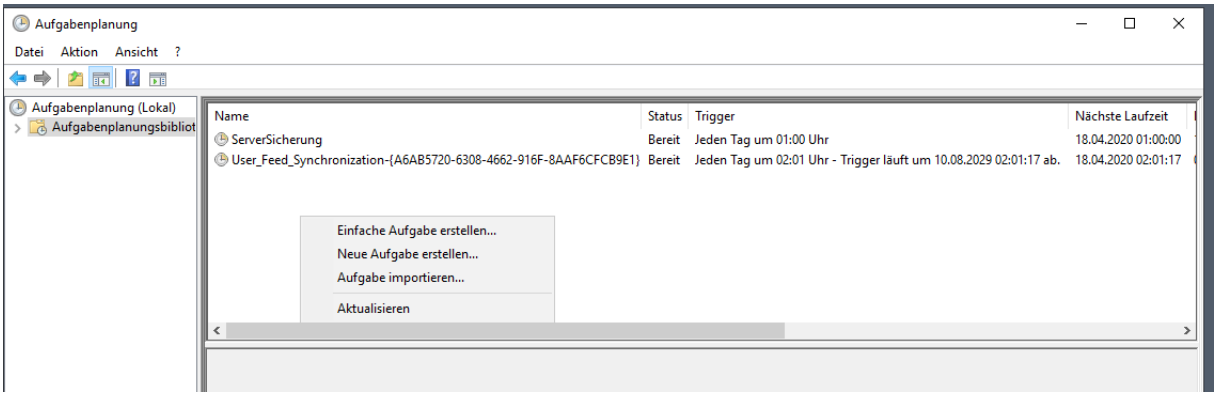
Damit ich auch nach den 119 Testtagen Freude am neuen Exchange Server habe, trage ich den Produktschlüssel ein. Der Vorgang muss mit einem Neustart des Information Store Service abgeschlossen werden. In größeren Umgebungen sollte der Prozess daher VOR der Verschiebung der Mailboxen stattfinden.

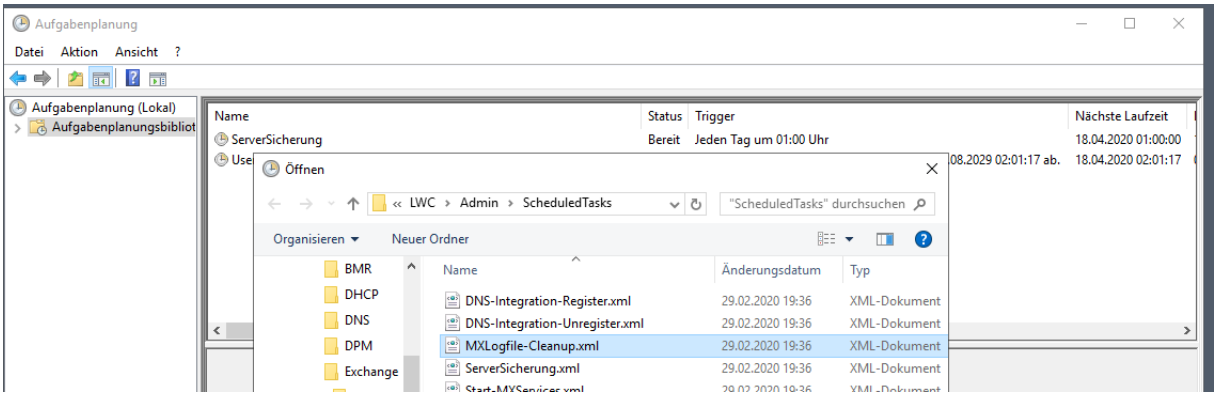
```
249      # Aktivierung
250          Set-ExchangeServer -ProductKey '[redacted]' -Identity "WS-MX2"
251          Invoke-Command -ComputerName "WS-MX2" -ScriptBlock { Restart-Service -Name MSExchangeIS }
252
```

```
PS C:\>         Set-ExchangeServer -ProductKey '[redacted]' -Identity "WS-MX2"
WARNUNG: Der Product Key wurde überprüft und die Product ID erfolgreich erstellt. Diese Änderung ist erst nach einem Neustart
des Informationsspeicherdiensts wirksam.

PS C:\>         Invoke-Command -ComputerName "WS-MX2" -ScriptBlock { Restart-Service -Name MSExchangeIS }
WARNUNG: Warten auf Beendigung des Diensts "Microsoft Exchange Information Store (MSExchangeIS)"...
WARNUNG: Warten auf Start des Diensts "Microsoft Exchange Information Store (MSExchangeIS)"...

PS C:\>
```

Logfile-Optimierung

Jetzt möchte ich noch die Log-Optimierung platzieren. Ein Exchange Server protokolliert den ganzen Tag in etliche Logfiles. Jedes Log kann dabei mit einer Rotation und einer Bereinigung konfiguriert werden. Das ist mir echt zu aufwendig. Daher erstelle ich lieber einen Task, der auf der gesamten Systemplatte nach bestimmten Dateitypen sucht, die älter als ein Schwellwert sind. Gefundene Files werden dann gelöscht. Den Task importiere ich mit einer XML-Datei:



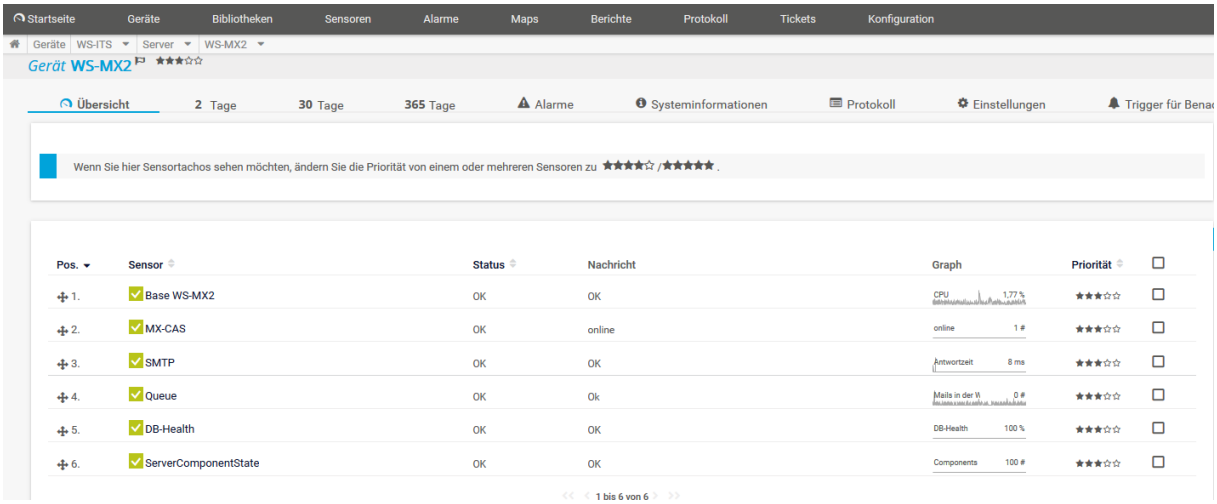


Das hier ist der Aufruf in der geplanten Aufgabe. Alle Logfiles (auch die des IIS), die älter sind als 14 Tage, werden gelöscht:

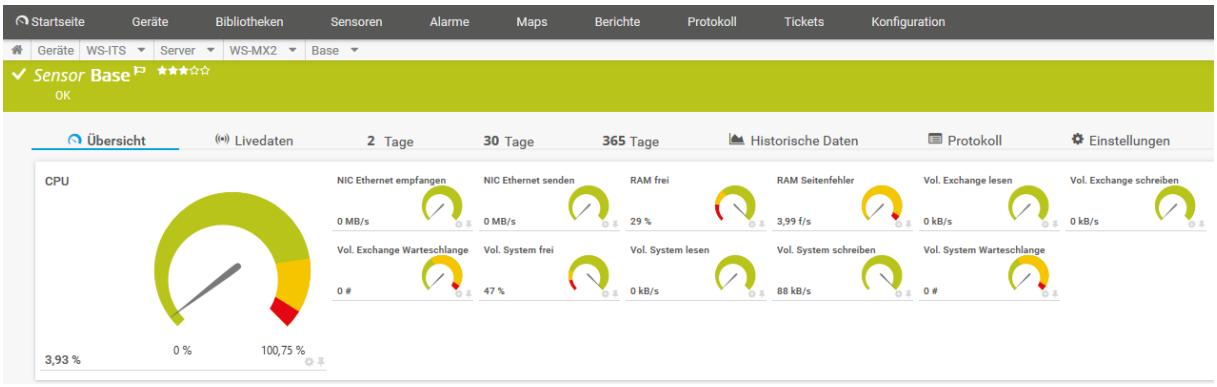
```
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe "& {Get-ChildItem -Path 'C:\Program
Files\Microsoft\Exchange Server\V15\Logging','C:\inetpub\logs\LogFiles' -Include
'*.log','*.bak','*.blg' -Recurse | Where-Object { $_.LastWriteTime -le (Get-Date).AddDays(-14) } |
Remove-Item -Confirm:$false -ErrorAction SilentlyContinue}"
```

Konfiguration des Monitorings

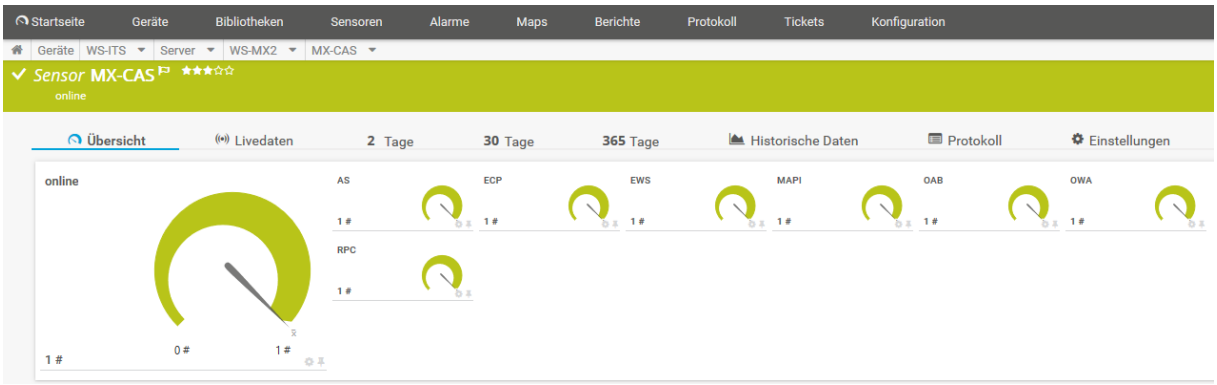
Zu den Nacharbeiten gehört auch das Monitoring. Diese Aufgabe übernimmt mein PRTG. Einige Sensoren habe ich bereits erstellt.



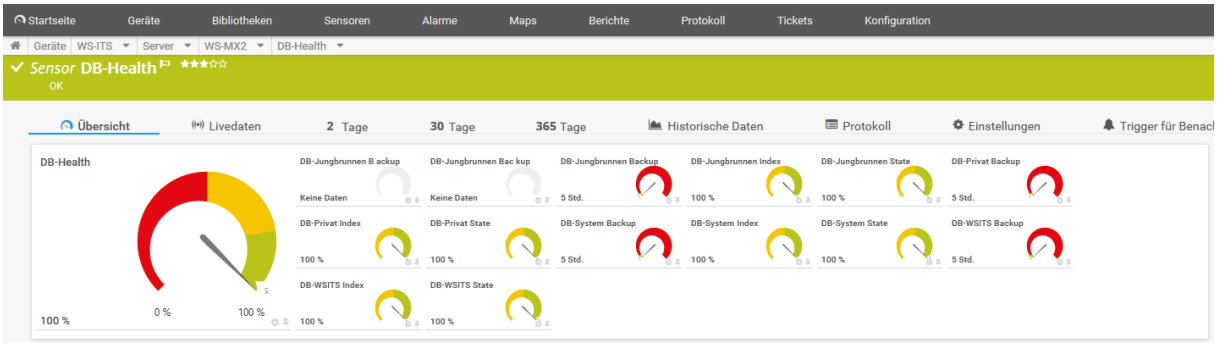
Dazu zählt auch mein selbstgeschriebener PowerShell-Script-Sensor „BASE“. Mit diesem kann ich die typischen Werte des Betriebssystems überwachen:



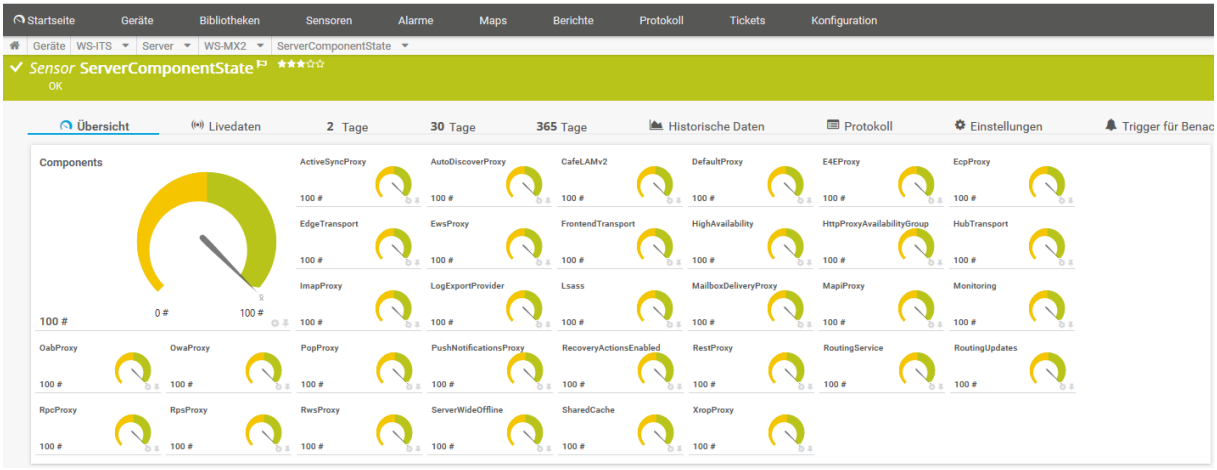
Der von mir geschriebene Sensor „MX-CAS“ überwacht die einzelnen Webservices des Client-Access-Services:



Ebenfalls selbst geschrieben überwacht der Sensor „DB-Health“ den Zustand der Datenbanken. Und im Vergleich zu dem Standard-Sensor von PRTG kann er auch mit dem neuen Indizierungsmechanismus vom Exchange Server 2019 umgehen:



Und zuletzt liefert mir ein selbst programmierter Sensor alle ServerComponentStates des Servers:



Zusammenfassung

Der erste Mailserver ist neu installiert. Mit allen Begleitdiensten, wie dem Backup und dem Monitoring war es viel Arbeit. Aber es gab bis auf die Probleme bei der Deinstallation des alten Servers keine großen Schwierigkeiten. Also kann es bald mit dem anderen Exchange Server weitergehen.